Async Programming Constructs in ES2017

What's the hype?

Ying Ka Ho

@yingkh_tweets

async/await

in ES2017

What's the hype?

Ying Ka Ho

@yingkh_tweets

Async function declaration:

```
async[no LineTerminator here]functionBindingIdentifier[?Yield, ?Await](FormalParameters[~Yield, ?Await]){
     AsyncFunctionBody
}

[+Default]async[no LineTerminator here]function(FormalParameters[~Yield, ?Await]){AsyncFunctionBody}
```

Await expression:

```
     await UnaryExpression[?Yield, +Await]
```

https://www.ecma-international.org/ecma-262/8.0/index.html

AJAX

- Event handlers
- Callbacks

Messy!

Readable?

Promise

- Callbacks [`promise.then(function)`]
- Generators [`yield`]

Javascript Generators

“Only old-school async requires you to write your code ‘inside out’. ”

- Stephen Cleary, who wrote his first async program in C++ in year 2001.

https://blog.stephencleary.com/2012/09/an-async-horror-story.html

Typical async function in js

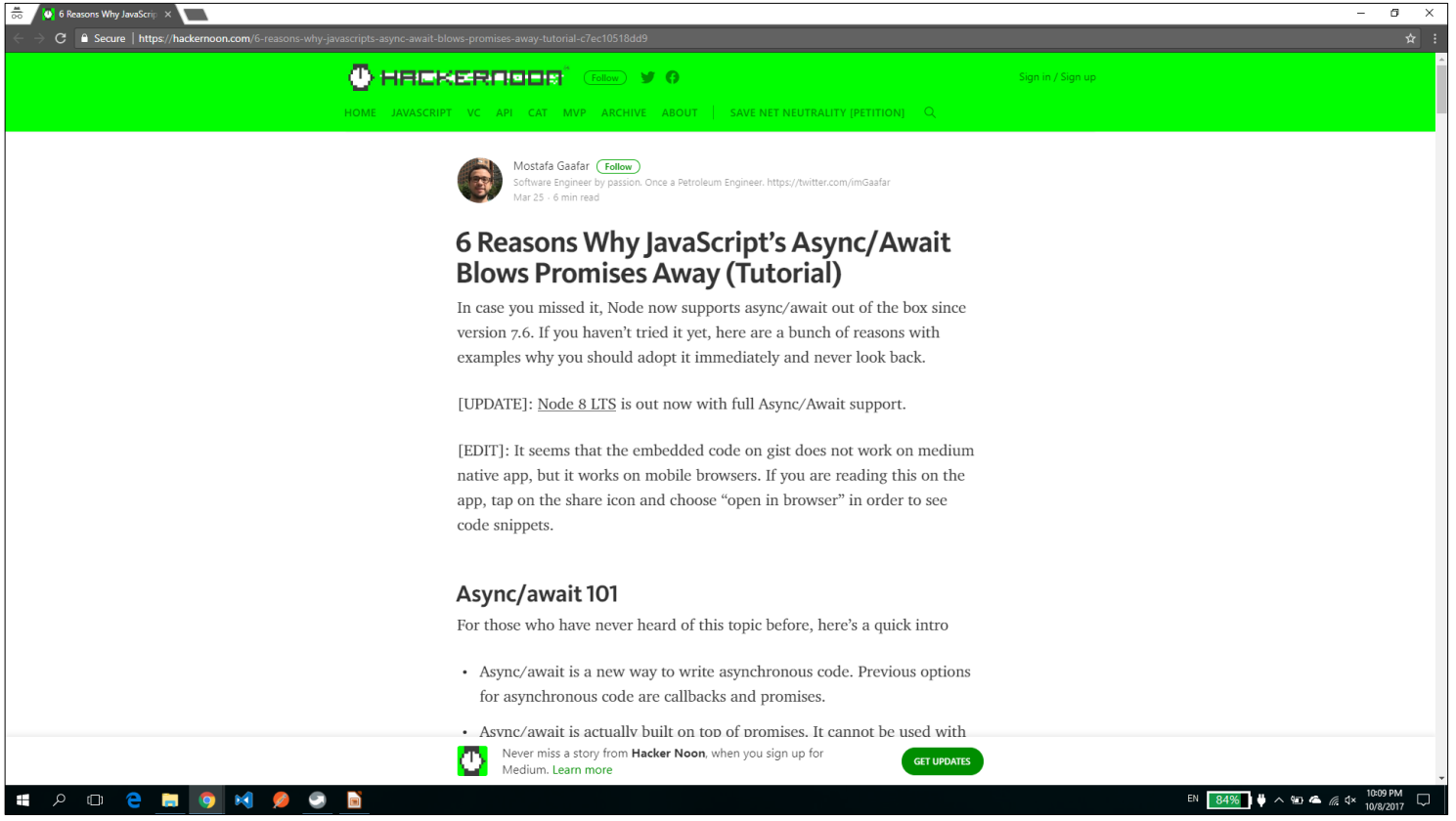
```
async function callFunctionDelayed(){
    var response = await getRandomQuotesDelayed();
    document.getElementById('content').innerHTML = response;
}
```

Typical async function in Typescript

```
async function printDelayed(elements: string[]) {
    for (const element of elements) {
        await delay(200);
        console.log(element);
    }
}

function delay(milliseconds: number) {
    return new Promise<void>(resolve => {
        setTimeout(resolve, milliseconds);
    });
}
```

https://www.typescriptlang.org/docs/handbook/release-notes/typescript-1-7.html



- Error handling
- Conditionals
- Intermediate values
- Error stacks
- Debugging

https://hackernoon.com/6-reasons-why-javascripts-async-await-blows-promises-away-tutorial-c7ec10518dd9

From “Harry Potter and the Philosopher’s Stone” by JK Rowling

HOGWARTS SCHOOLS OF WITCHCRAFT AND WIZARDRY

….

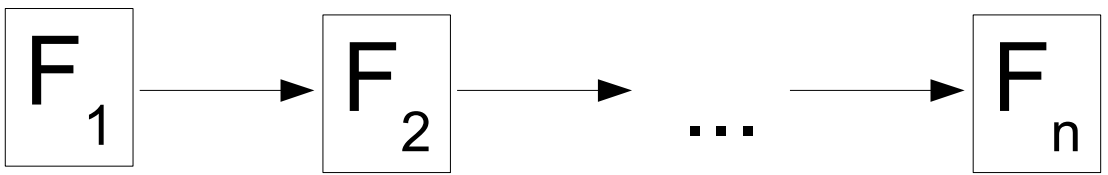
Terms begins on 1 September. We await your owl by no later than 31 July

Yours Sincerely,

Minerva McGonagall
Deputy Headmistress

Javascript

```
async function callFunctionDelayed(){
    var response = await getDataFromServerDelayed();
    var result = computeForResult(response);
    return result;
}
```



C#

```
public async Task<string> CallFunctionAsync(){
    var response = await GetDataFromServerAsync();
    var result = ComputeForResult(response);
    return result;
}
```

Bad Examples

```
async function getRandomQuotesDelayed(){
    var request = new XMLHttpRequest();
    return await new Promise(function(resolve, reject){
        request.onreadystatechange = function() {
            if (request.readyState === XMLHttpRequest.DONE){
                if (request.status === 200){
                    resolve(request.responseText);
                }
                else{
                    reject(new Error('There is a problem in requesting for a webpage.'));
                }
            }
        }
        request.open("GET", 'http://ron-swanson-quotes.herokuapp.com/v2/quotes');
        request.send();
    });
}
```

Bad Examples

```
async function computeDelayed(){
    var result = synchronousComputations();

    return await new Promise(function(resolve, reject){
            resolve(result);
    });
}
```

```
async function callAsyncFunctionDelayed(){
    synchronousWork();
    var result = await asyncFuncDelayed();
    return result;
}
```

Notes on async/await

- The understanding of the meaning of ‘await’ to prevent the misuse of the keyword.
- Using ‘async’ keyword when you need to return the result in an asynchronous function.

Notes on learning async programming

- Don’t just stop on async/await.
- There’s reactive programming, actor programming, parallel programming and dataflow programming
- The plot thickens when more than one threads are involved.
- Don’t forget about how useful is async on IO.

Summary

- Async programming constructs before ES2017
- async/await constructs
- Notes on async programming.