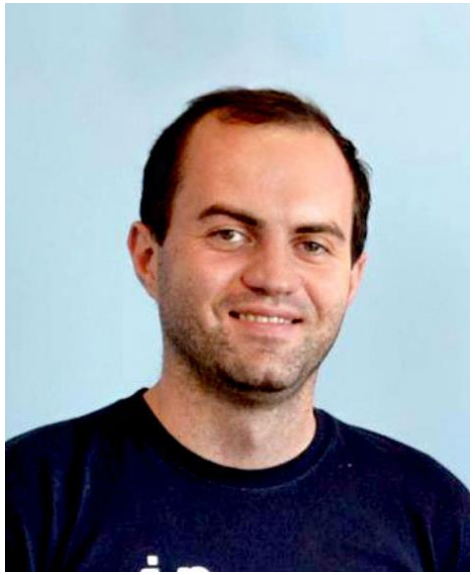
JavaScript Базовый

Асинхронный код. Promise

Introduction



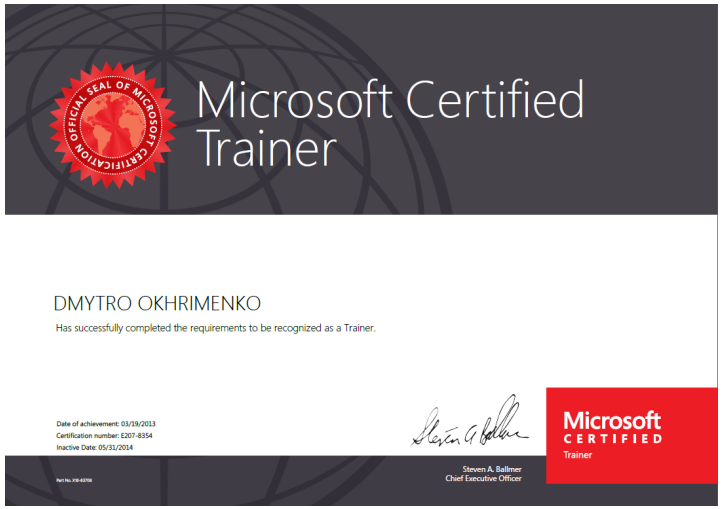
Охрименко Дмитрий

MCT

_okhrimenko

dmitriy.okhrimenko

dokhrimenko



MCID: 9210561

Тема урока

Асинхронный код. Promise

План урока

Синхронный и асинхронный код
Функции обратного вызова для асинхронного кода
Promise
Promise API

Синхронный код

Синхронный код - код, выполняющийся последовательно, каждая операция ожидает завершение предыдущей.

```
function download() {
    скачивает и возвращает данные
    выполняется ~10 секунд
}
```

```
download();   <- ~10 сек.
download();   <- ~10 сек.
download();   <- ~10 сек.

// другие вычисления
foo();        <- выполнится через ~30 сек.
```

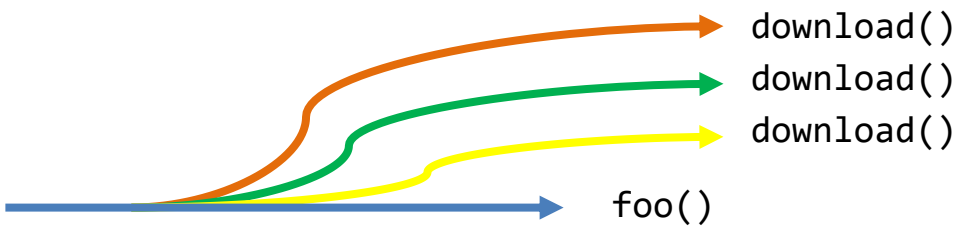
Асинхронный код

Асинхронный код – код, выполняющийся параллельно, а не последовательно. В большинстве случаев асинхронное выполнение кода подразумеват выделение новых ресурсов для выполнения асинхронных вызовов – потоков.

```
download();
download();
download();

// другие вычисления
foo();
```

Не дожидаемся завершения и переходим к следующей операции.
Не дожидаемся завершения и переходим к следующей операции.
Не дожидаемся завершения и переходим к следующей операции.
Выполнится до того, как закончится выполнение трех методов download



Организация асинхронного кода

Варианты организации асинхронного кода:
- Callback (функция, обратного вызова)
- Promise
- Шаблон Observer

```
function callback(result) {
   ...
}

download(callback);
```


```
let promise = download();

promise.then(callback);
```

```
let observable = download();

observable.subscribe(callback);
```

Promise

Promise – объект, который хранит конечный результат отложенной операции. Promise – представляет значение, которое еще не существует. Daniel P. Friedman и David Wise предложили термин Promise в 1976 году.

Возможные состояния объекта promise:

- Fulfilled
- Rejected
- Pending

Может переходить из состояния pending либо в fulfilled либо в rejected.

`p.then(f, r)` – если p в состоянии **`fulfilled`** функция f будет вызвана.

`p.then(f, r)` – если p в состоянии **`rejected`** функция r будет вызвана.

Во всех остальных случаях p в состоянии pending.

`settled` – promise перешел в состояние `rejected` или `fulfilled`

Итоги

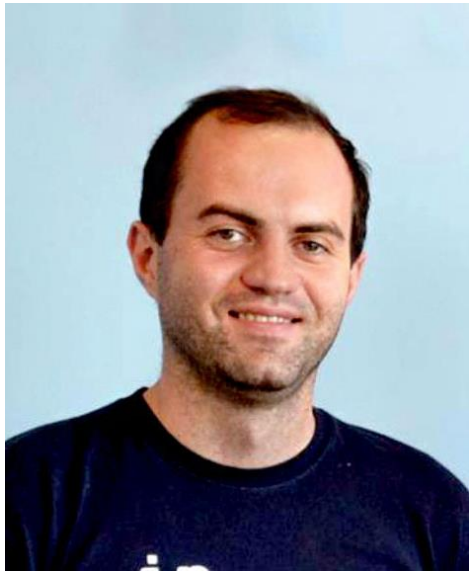
Синхронный код – операции выполняются последовательно
Асинхронный код – операции выполняются параллельно

Способы обработки асинхронной операции – **callback**, **promise**, **observer**

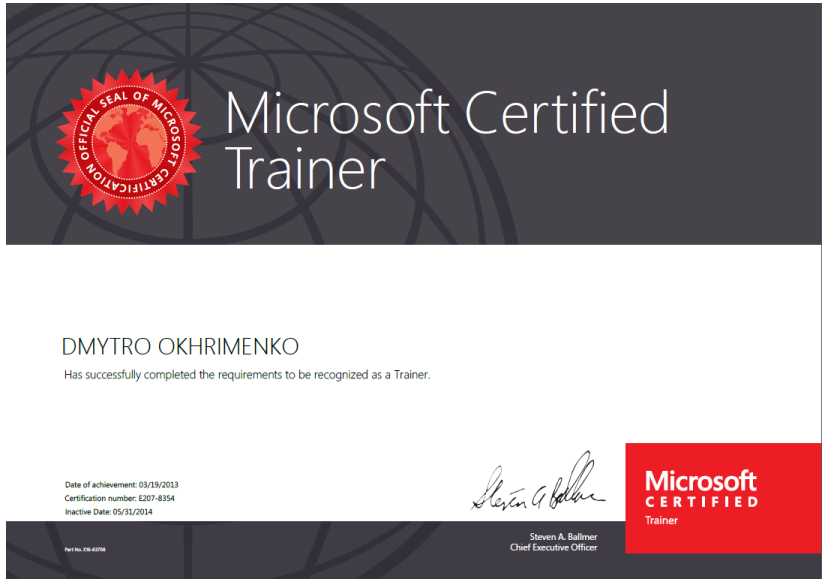
Promise – объект, представляющий **результат асинхронной операции**
Основные методы promise – **then**, **catch**, **finally**

Если асинхронные операции должны выполняться одна за другой, промисы можно выстроить в **цепочку**

Спасибо за внимание! До новых встреч!



Охрименко Дмитрий
MCT



MCID: 9210561

Информационный видеосервис для разработчиков программного обеспечения