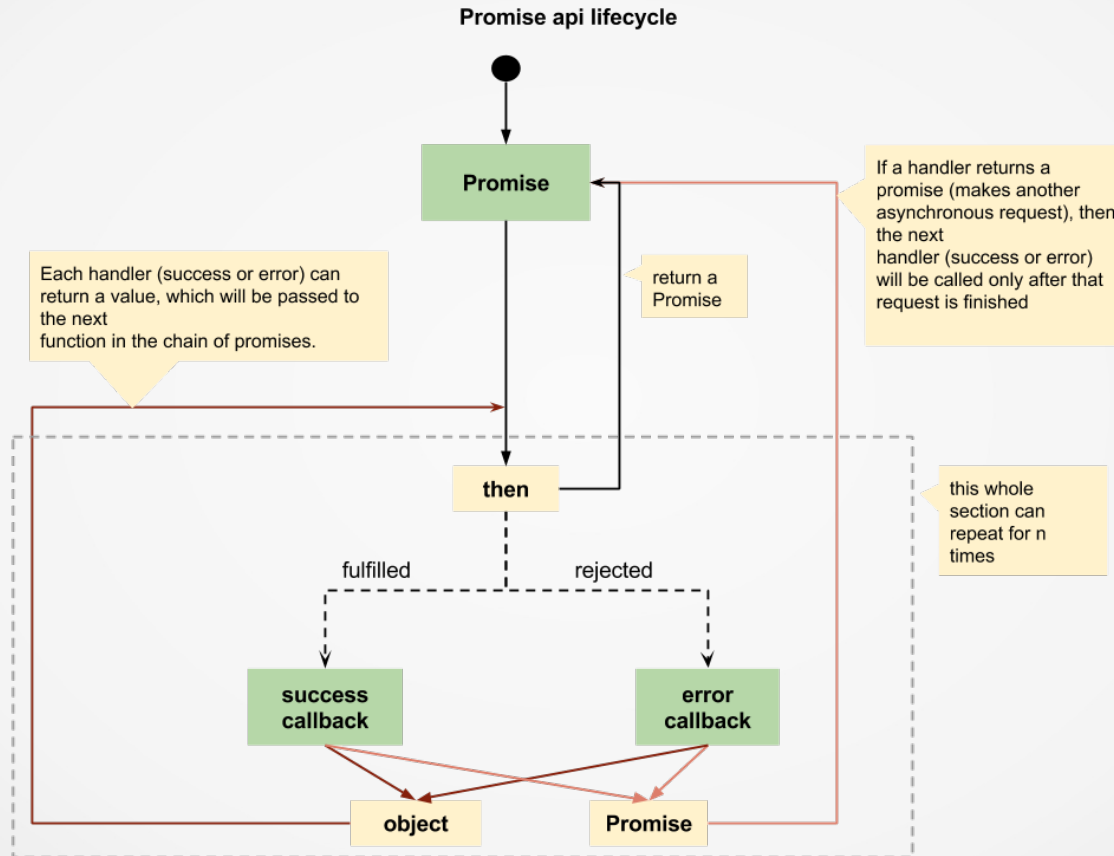


## Определение

**Promise** – это специальный объект, который содержит своё состояние. Вначале pending («ожидание»), затем – одно из: fulfilled («выполнено успешно») или rejected («выполнено с ошибкой»).

# Диаграмма состояния promise



## Синтаксис создания Promise

```
var p = new Promise(  
  function(resolve, reject){  
    ...  
    if(something)  
      resolve({});  
    else{  
      reject(new Error());  
    }  
  })  
);  
  
p.then(  
  function(data){  
    ...  
  },  
  function(err){  
    ...  
  }  
);
```

The diagram illustrates the mapping between the `resolve` and `reject` methods of a Promise constructor and the corresponding callback functions in the `then` method. Two curved arrows show the flow: one from `resolve({})` to the first callback `function(data){ ... }`, and another from `reject(new Error());` to the second callback `function(err){ ... }`.

# Синтаксис создания Promise

```
var promise = new Promise(function(resolve, reject) {  
  // здесь пишем логику, потом...  
  if (/* ..если всё закончилось успехом */ ) {  
    resolve("Работает!");  
  } else {  
    reject(Error("Сломалось")); }  
});  
  
promise.then(function(result){console.log(result); //  
  "Обрабатываем результат!" }  
  , function(err) { console.log(err); // Ошибка: "Сломалось"  
});
```

# Промисификация

**Промисификация** – это когда берут асинхронный функционал и делают для него обёртку, возвращающую промис.

## Навешиваем цепочку Promise

```
getUser(connection).then(function (result) {  
    res.render('admin', { table: result });  
    return "AAA";  
}).then(function (result) {  
    console.log(result);  
}).catch(function () {  
    console.log("Promise Rejected");  
});
```

## Список литературы

- <https://learn.javascript.ru/promise>
- <https://habrahabr.ru/post/209662/>



Когда даешь пустые обещания...

```
new Promise(() => {});
```