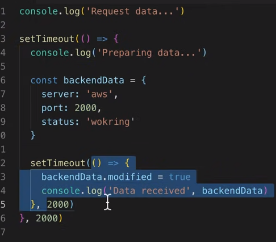
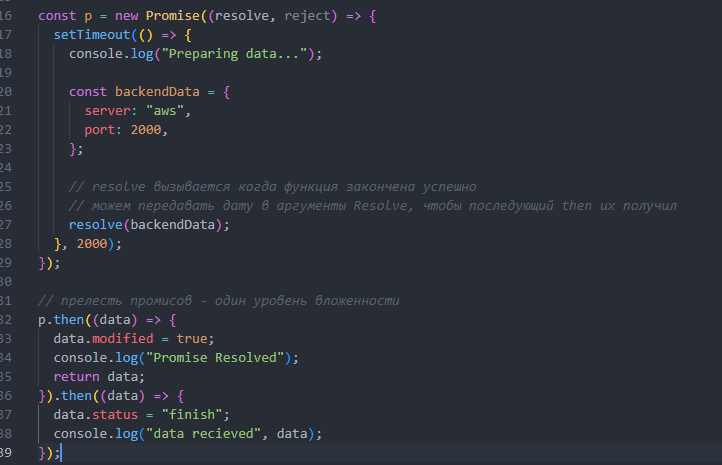
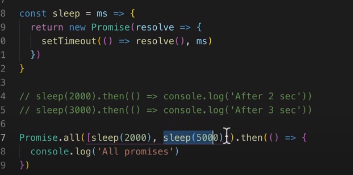
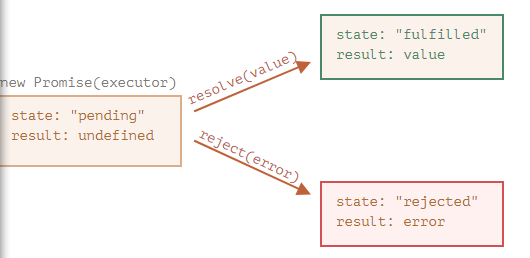
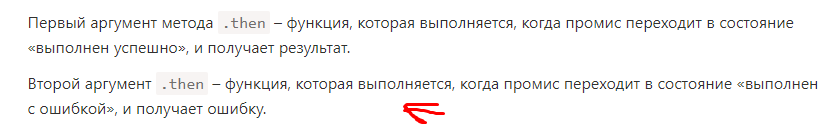
**1. Промисы ES6.  
  
1.1.**Для чего? Для избежания callback hell’а.  
До них нужно было прописывать много callback’ов для управления данными пришедшими асинхронно.  
  
  
  
Промисы хороши одним уровнем вложенности.  
Промис можно создать через new Promise(resolve, reject).  
Обработка промисов: then, catch, finally.  
  
  
  
  
У глобального класса Promise присутствуют ещё 2 метода: all и race.  
В all передаём массив промисов и возвращаем промис, который будет выполнен, когда ВСЕ промисы завершатся.  
  
  
  
В Race передаём также массив промисов и возвращаем промис, который выполнится при завершении быстрейшего промиса.  
  
  
**1.2.**  
Промис – что-то что случится в будущем, но с задержкой.  
Пока данные не получены, fetch возвращает промис.  
  
  
<https://learn.javascript.ru/promise>  
  
https://habr.com/ru/post/439746/  
  
https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global\_Objects/Promise  
  
<https://www.youtube.com/watch?v=83vNg357dBM&ab_channel=%D0%95%D0%BB%D0%B5%D0%BD%D0%B0%D0%9B%D0%B8%D1%82%D0%B2%D0%B8%D0%BD%D0%BE%D0%B2%D0%B0%E2%80%94%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%BE%D0%B2%D0%B5%D0%B1-%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8>  
  
<https://www.youtube.com/watch?v=Sr0WT-XHwd0&t=251s&ab_channel=WebDev%D1%81%D0%BD%D1%83%D0%BB%D1%8F.%D0%9A%D0%B0%D0%BD%D0%B0%D0%BB%D0%90%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%9B%D1%83%D1%89%D0%B5%D0%BD%D0%BA%D0%BE>  
  
<https://www.youtube.com/watch?v=kdJKuDLuOkY&ab_channel=LISAWEBGIRL>  
  
<https://www.youtube.com/watch?v=DHvZLI7Db8E&ab_channel=WebDevSimplified>  
  
<https://www.youtube.com/watch?v=PoRJizFvM7s&ab_channel=TraversyMedia>  
  
<https://www.youtube.com/watch?v=cuEtnrL9-H0&ab_channel=WebDevSimplified>  
  
<https://www.youtube.com/watch?v=s6SH72uAn3Q&ab_channel=techsith>  
  
<https://www.youtube.com/watch?v=_8gHHBlbziw&ab_channel=developedbyed>  
  
  
  
У объекта промис есть внутренние свойства:  
1) state: сначала pending, потом fulfilled (при вызове resolve), или Rejected (при reject).  
2) result – сначала udefined, потом value при resolve(value) или error при reject(error).  
  
  
  
  
Типо можно вообще без catch работать.  
  
  
  
  
  
  
  
  
  
  
  
  
  
JavaScript — это однопоточный язык программирования  
