# 实现简化的 promise

## 代码

function Promise(func) {  
 this.fullfilled = false;  
 this.rejected = false;  
 this.pending = true;  
 this.handlers = [];  
 this.errorHandlers = [];  
 function resolve(...args) {  
 this.handlers.forEach(handler => handler(...args));  
 }  
 function reject(...args) {  
 this.errorHandlers.forEach(handler => handler(...args));  
 }  
 func.call(this, resolve.bind(this), reject.bind(this));  
}  
  
Promise.prototype.then = function(func) {  
 this.handlers.push(func);  
 return this;  
};  
Promise.prototype.catch = function(func) {  
 this.errorHandlers.push(func);  
 return this;  
};  
  
Promise.race = promises =>  
 new Promise((resolve, reject) => {  
 promises.forEach(promise => {  
 promise.then(resolve, reject);  
 });  
 });  
  
Promise.all = promises =>  
 new Promise((resolve, reject) => {  
 let len = promises.length;  
 let res = [];  
 promises.forEach((p, i) => {  
 p.then(r => {  
 if (len === 1) {  
 resolve(res);  
 } else {  
 res[i] = r;  
 }  
 len--;  
 }, reject);  
 });  
 });  
  
// test  
const p1 = new Promise(resolve =>  
 setTimeout(resolve.bind(null, "resolved"), 3000)  
);  
p1.then(console.log).then((...args) => console.log("second", ...args));  
  
const p2 = new Promise((resolve, reject) =>  
 setTimeout(reject.bind(null, "rejected"), 3000)  
);  
p2.then(console.log).catch((...args) => console.log("fail", ...args));