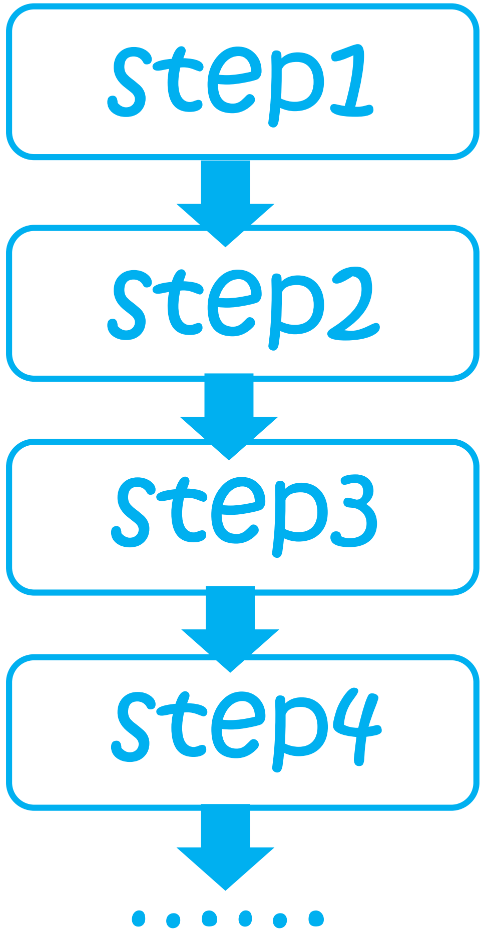
Promise

-- async is wonderful

One day 有一个项目，有很多异步的过程

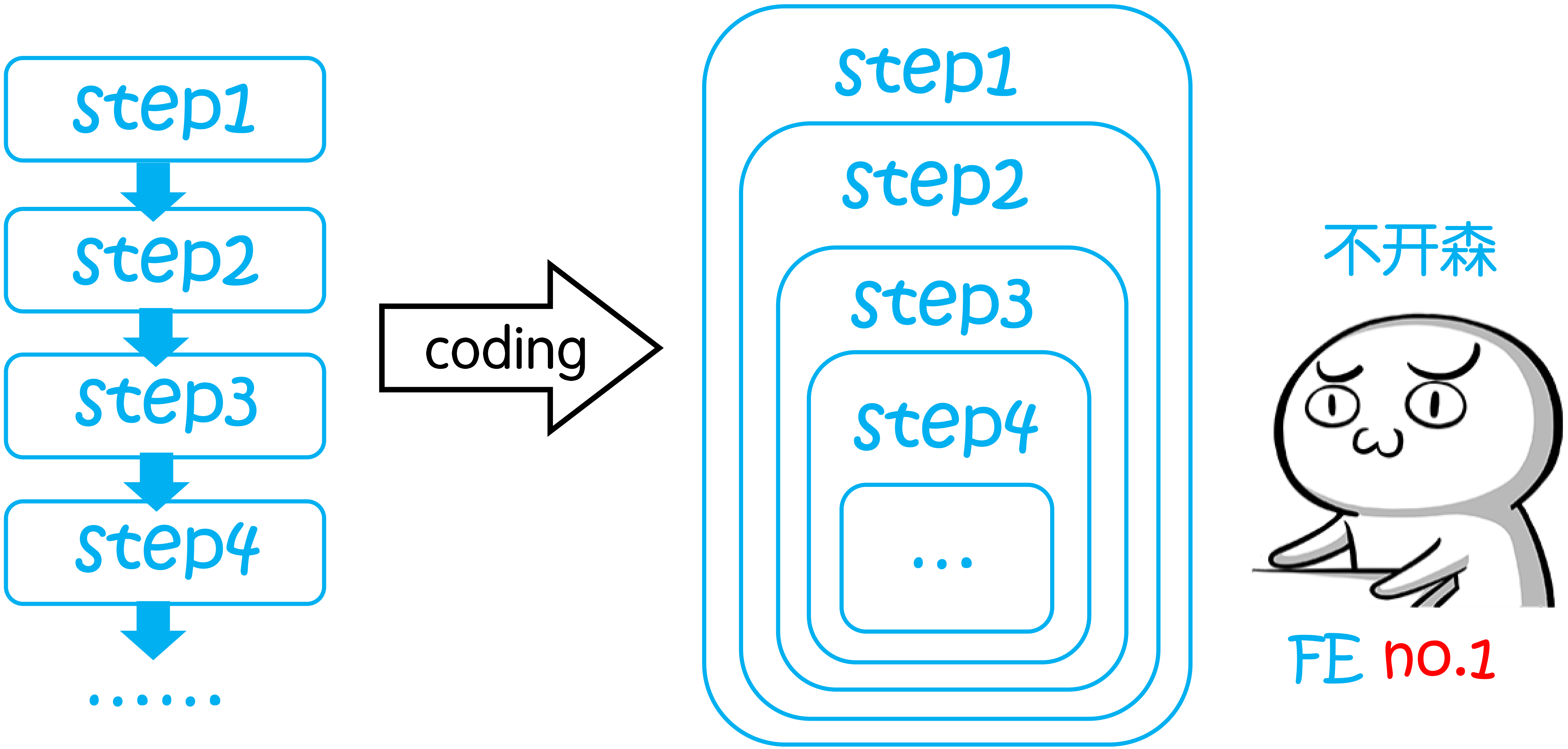
Design



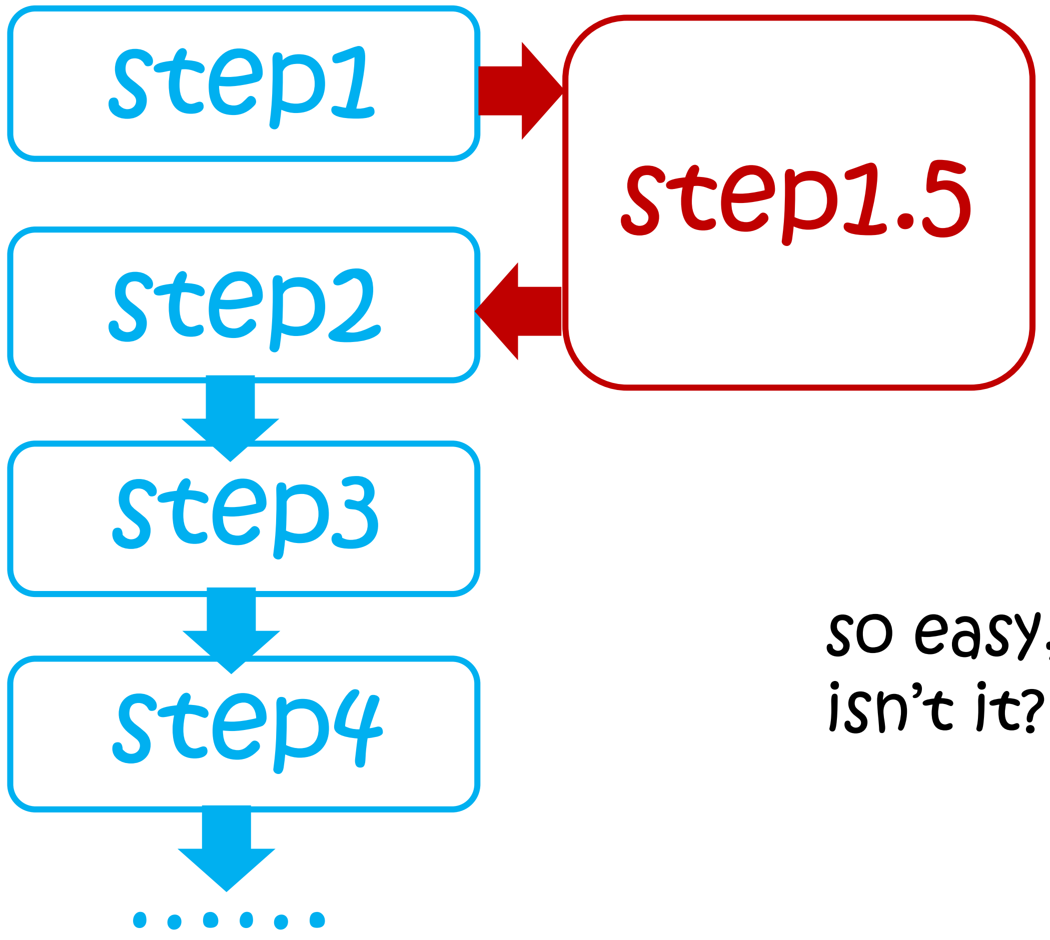
so easy

FE no.1

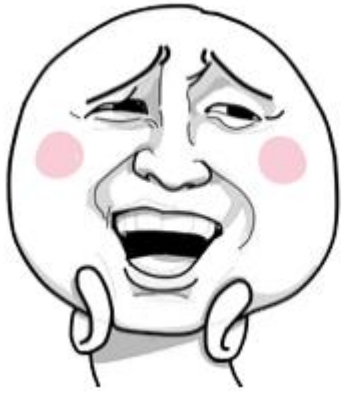
Coding 阶段


step1
step2
step3
step4
......
coding
step1
step2
step3
step4
...
不开森
FE no.1

二期增加需求



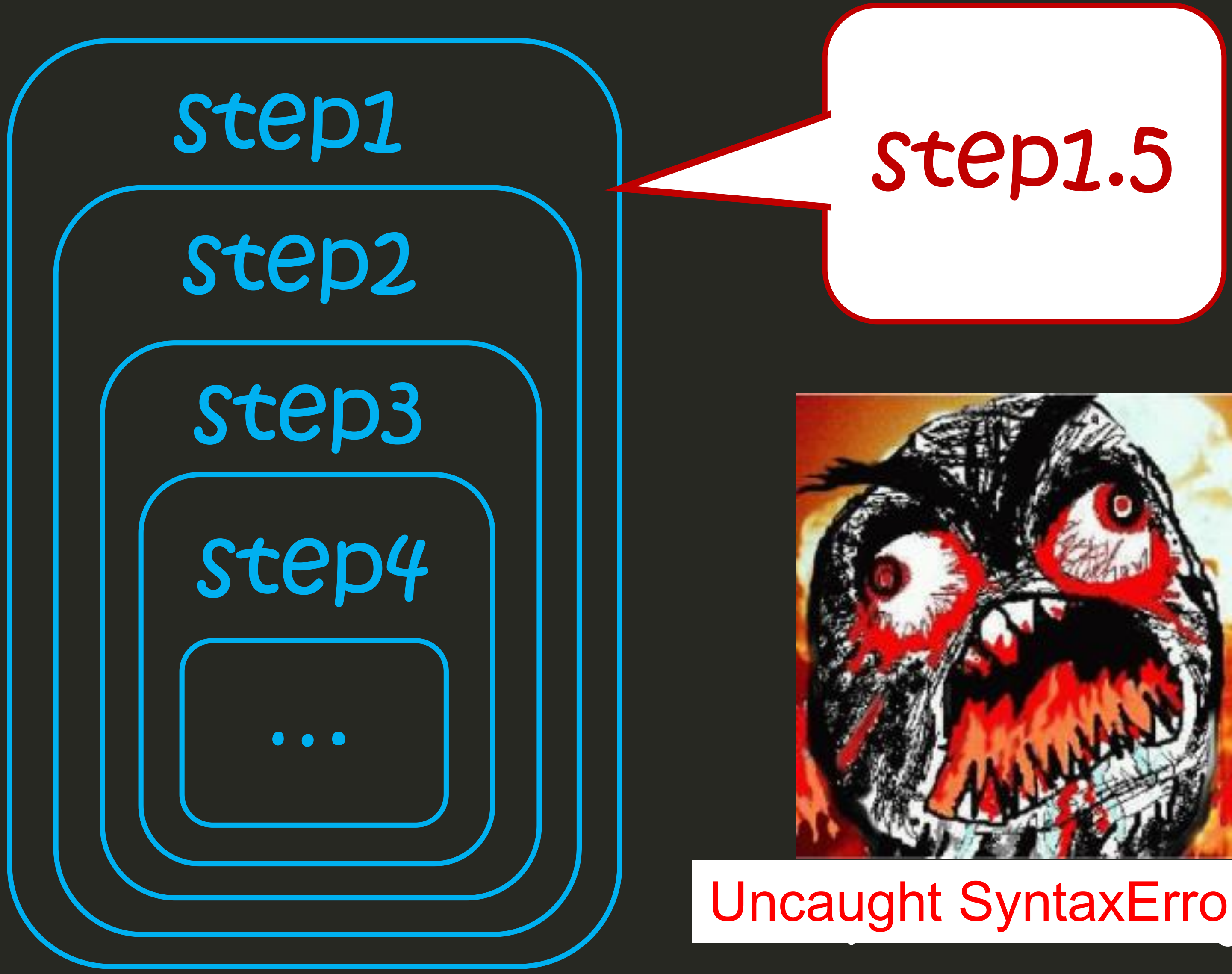
so easy,
isn't it?



PM

FE no.2

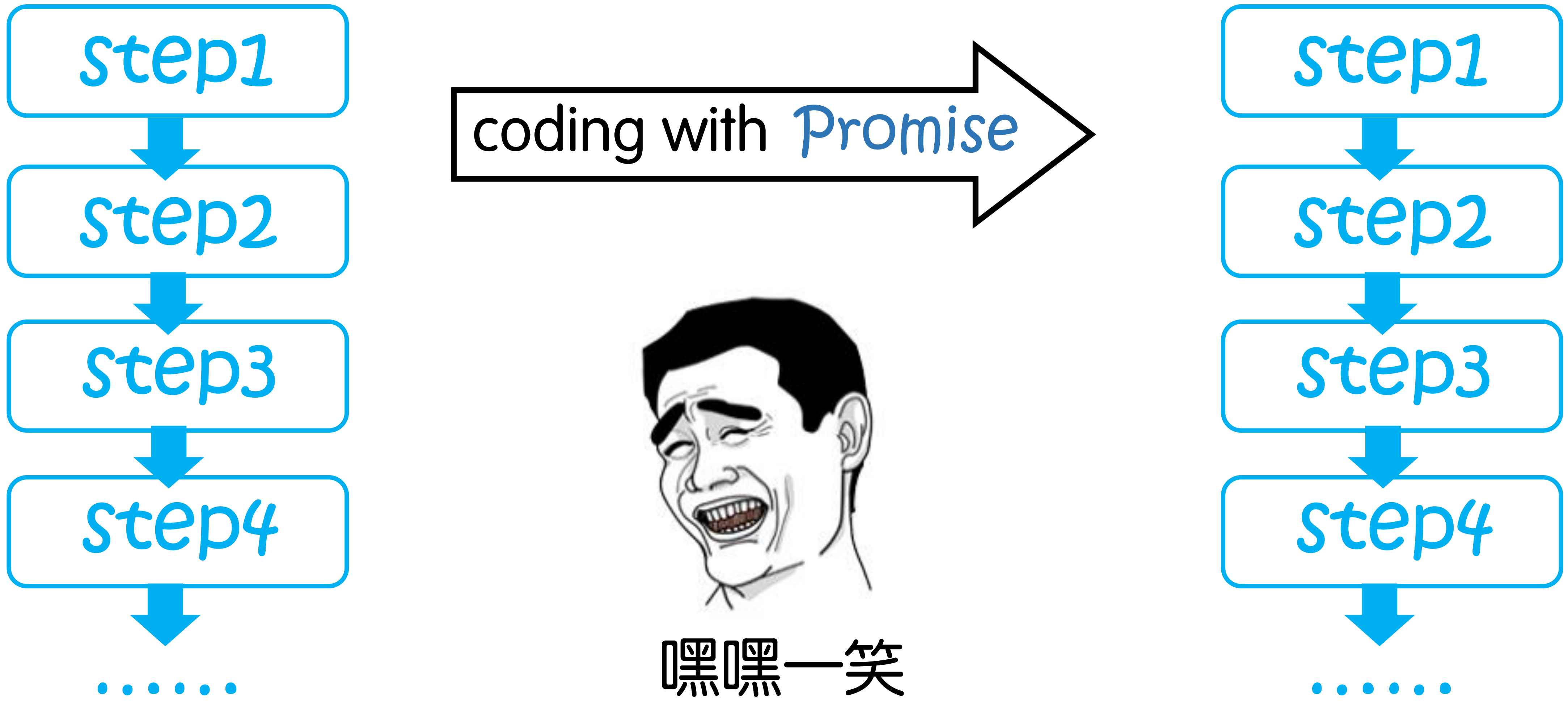
addrecord.js × chain_local_promise_loop_seq.js × chain_local_promise_loop.js × web.css × untitled ×



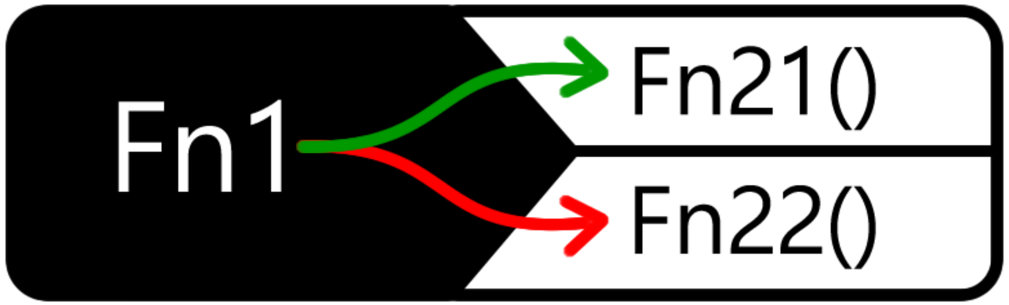
WTF

Uncaught SyntaxError: Unexpected token }

现在，有了Promise 就是这么愉快



Promise



Resolve() Reject()

Promise.then

Jquery实现的Promise

推荐文章

http://www.ruanyifeng.com/blog/2011/08/a_detailed_explanation_of_jquery_deferred_object.html

http://zsuczw.iteye.com/blog/1121129

```
var dtd = $.Deferred();
//实例一个deffered对象

dtd.promise(myFunc);
//关联这个Func（jquery自己的设计）

myFunc(){
        something async…
        if(成功){
                dtd.resolve(data);
        }else{
                dtd.reject(data);
        }
}
```

```
dtd.then(
        function(data){
                成功之后干的事情…
        },
        function(data){
                失败之后干的事情
        }
);
```

为了理解这个东西，这里需要问一个问题，这里myFunc执行了吗?

略施小计实现de链式

```
myFunc1(){
        something async…
        if(成功){
                dtd1.resolve(data);
        }else{
                dtd1.reject(data);
        }
        return dtd1 ;
}
```

```
myFunc2(){
        something async…
        if(成功){
                dtd2.resolve(data);
        }else{
                dtd2.reject(data);
        }
        return dtd2 ;
}
```

```
myFunc3(){
        something async…
        if(成功){
                dtd3.resolve(data);
        }else{
                dtd3.reject(data);
        }
        return dtd3 ;
}
```

```
myFunc1()
.then(function(){
        return myFunc2();
})
.then(function(){
        return myFunc3();
});
```

链式Pipe

```
myFunc1(){
        something async…
        if(成功){
                dtd1.resolve(data);
        }else{
                dtd1.reject(data);
        }
        return dtd1 ;
}
```

```
myFunc2(){
        something async…
        if(成功){
                dtd2.resolve(data);
        }else{
                dtd2.reject(data);
        }
        return dtd2 ;
}
```

```
myFunc3(){
        something async…
        if(成功){
                dtd3.resolve(data);
        }else{
                dtd3.reject(data);
        }
        return dtd3 ;
}
```

```
myFunc1()
.pipe(function(){
        return myFunc2();
})
.pipe(function(){
        return myFunc3();
});
```

合并promise -> 并行

```
$.when(promise1,promise2).then(..)
```

现代化的原生的promise

推荐文章 http://www.html5rocks.com/en/tutorials/es6/promises/

这篇文章结尾介绍了一个github项目用于在老浏览器里实现原生promise api

```
var promise = new Promise(function(resolve, reject) {
      // do a thing, possibly async, then…
      if (/* everything turned out fine */) {
            resolve("Stuff worked!");
      } else {
            reject("It broke"); }
      });
```

不一样的写法，一样思路

```
myFunc1(){
    var promise = new Promise(
        function(resolve, reject) {
         // do a thing async
         if (完成) {
              resolve(data);
          } else {
              reject(data);
          }
        });
    return promise;
}

myFunc2
myFunc3
```

串行

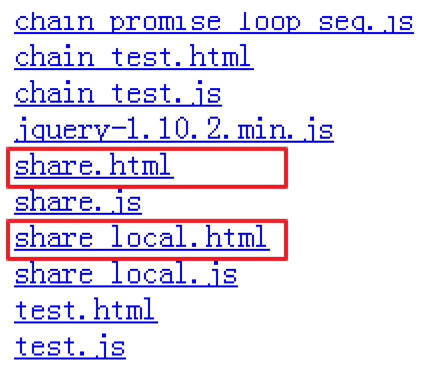
```
myFunc1()
.then(function(data){
            return myFunc2(data);
})
.then(function(data){
            return myFunc3 (data);
})
```

并行

```
Promise.all(arrayOfPromises).then(function(arrayOfResults) {
//...
});
```

包教包会de代码栗子

http://wangcongwu.fe.baidu.com/deferred/



chain_promise_loop_seq.js
chain_test.html
chain_test.js
jquery-1.10.2.min.js
share.html
share.js
share_local.html
share_local.js
test.html
test.js

3Q for watChing