1. debugger

After console.log, 'debugger;' is my favorite quick and dirty debugging tool. Once it's in your code, Chrome will automatically stop there when executing. You can even wrap it in conditionals, so it only runs when you need it.

在console.log之后，“调试器” 是我最喜欢的快速和肮脏的调试工具。 一旦它在您的代码中，Chrome会在执行时自动停止。 你甚至可以用条件包装它，所以它只在你需要的时候运行

<script>

var x=1;

if(x==2){

debugger;

alert('debugger over');

}

alert(x);

</script>

alert(x)还是会执行的 无论是否debugger

1. display objects as a table

Sometimes, you have a complex set of objects that you want to view. You can either console.log them and scroll through the list, or break out the console.table helper. Makes it easier to see what you’re dealing with!

有时，你有一个复杂的对象，你想要查看。 您可以console.log他们和滚动列表中，或打破了console.table帮手。 使你更容易看到你正在处理的事情！

var animals = [

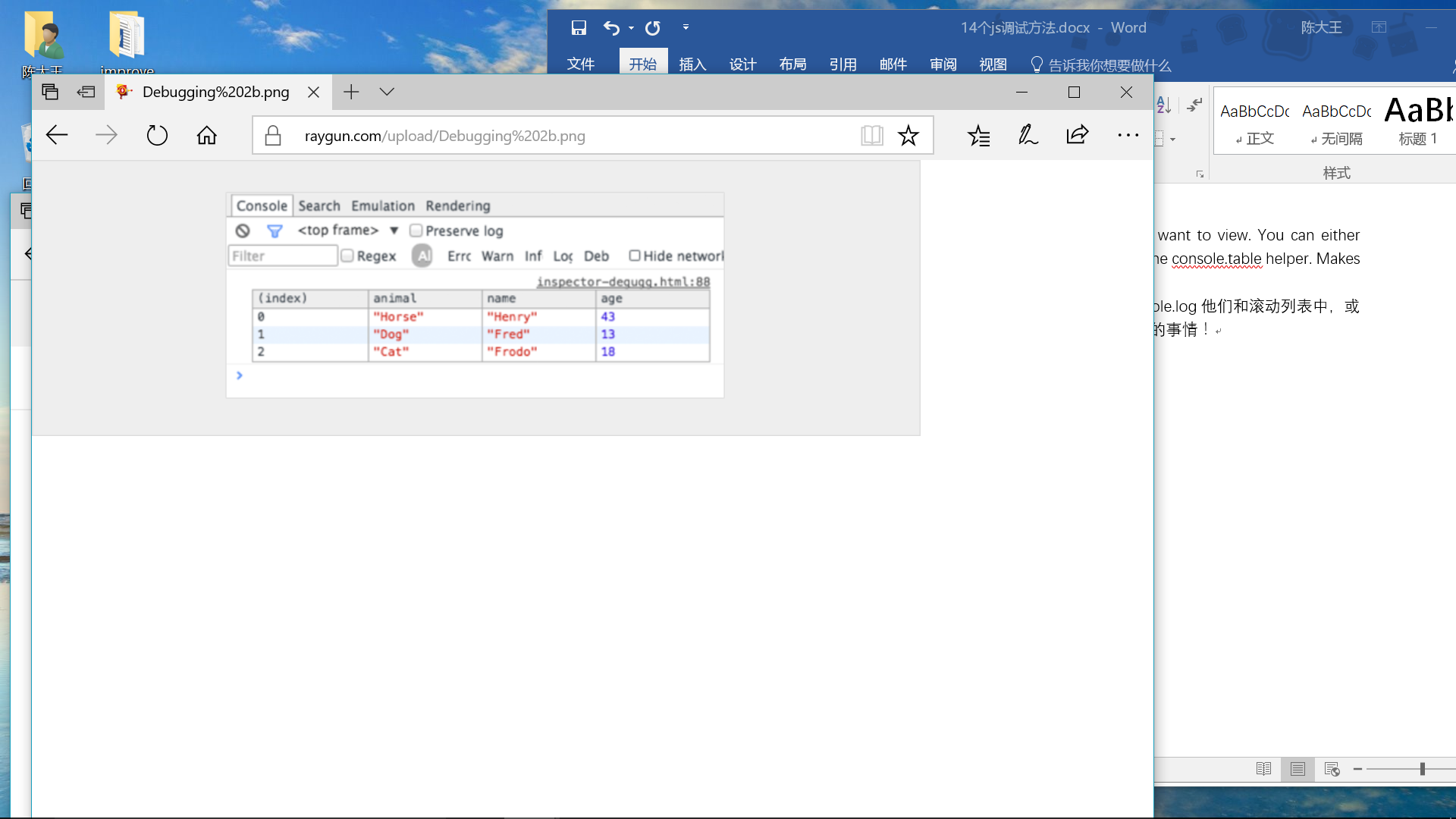
{ animal: 'Horse', name: 'Henry', age: 43 },

{ animal: 'Dog', name: 'Fred', age: 13 },

{ animal: 'Cat', name: 'Frodo', age: 18 }

];

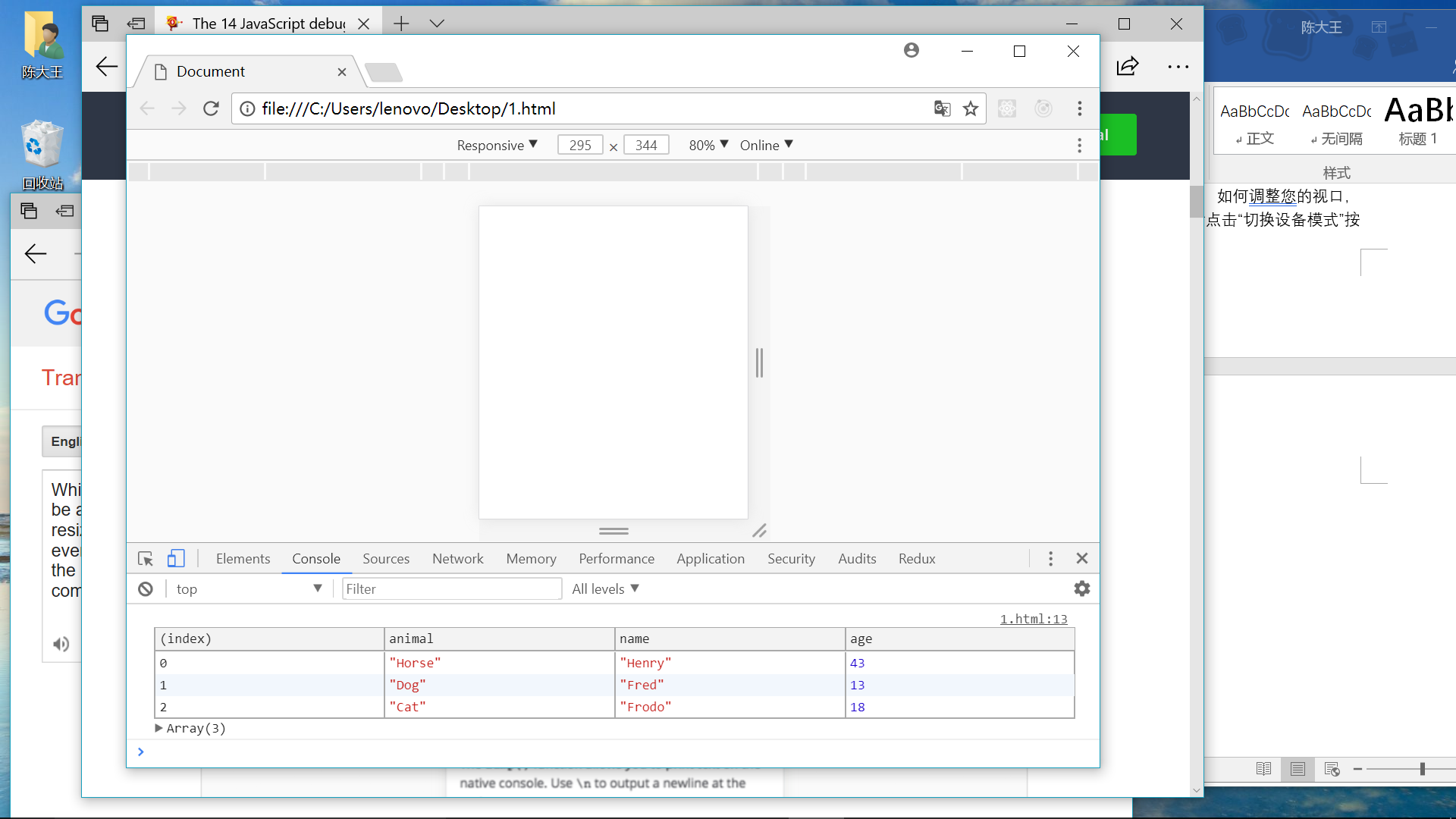
console.table(animals);



1. try all the sizes

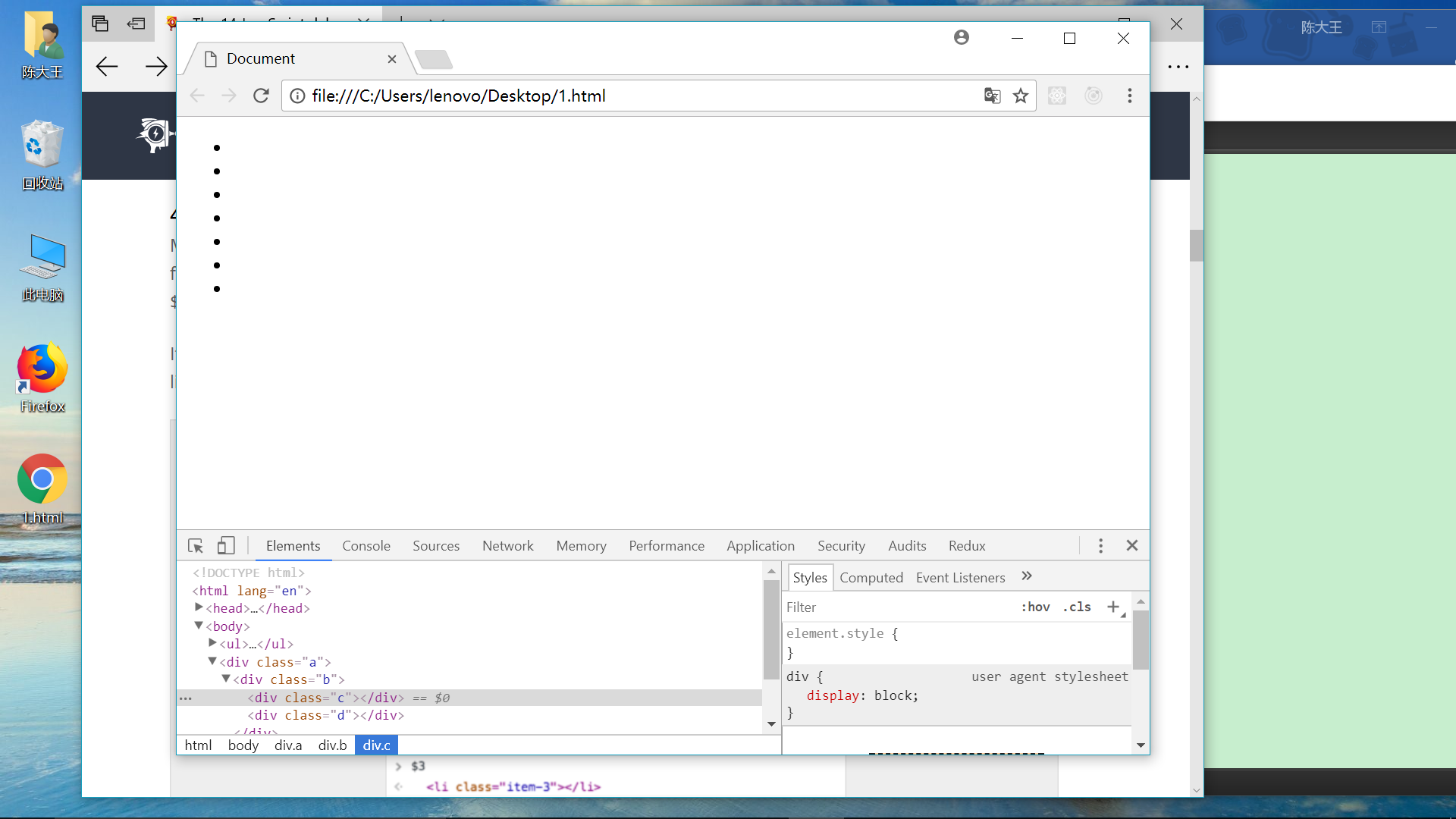
While having every single mobile device on your desk would be awesome, it’s not feasible in the real world. How about resizing your viewport instead? Chrome provides you with everything you need. Jump into your inspector and click the ‘toggle device mode’ button. Watch your media queries come to life!

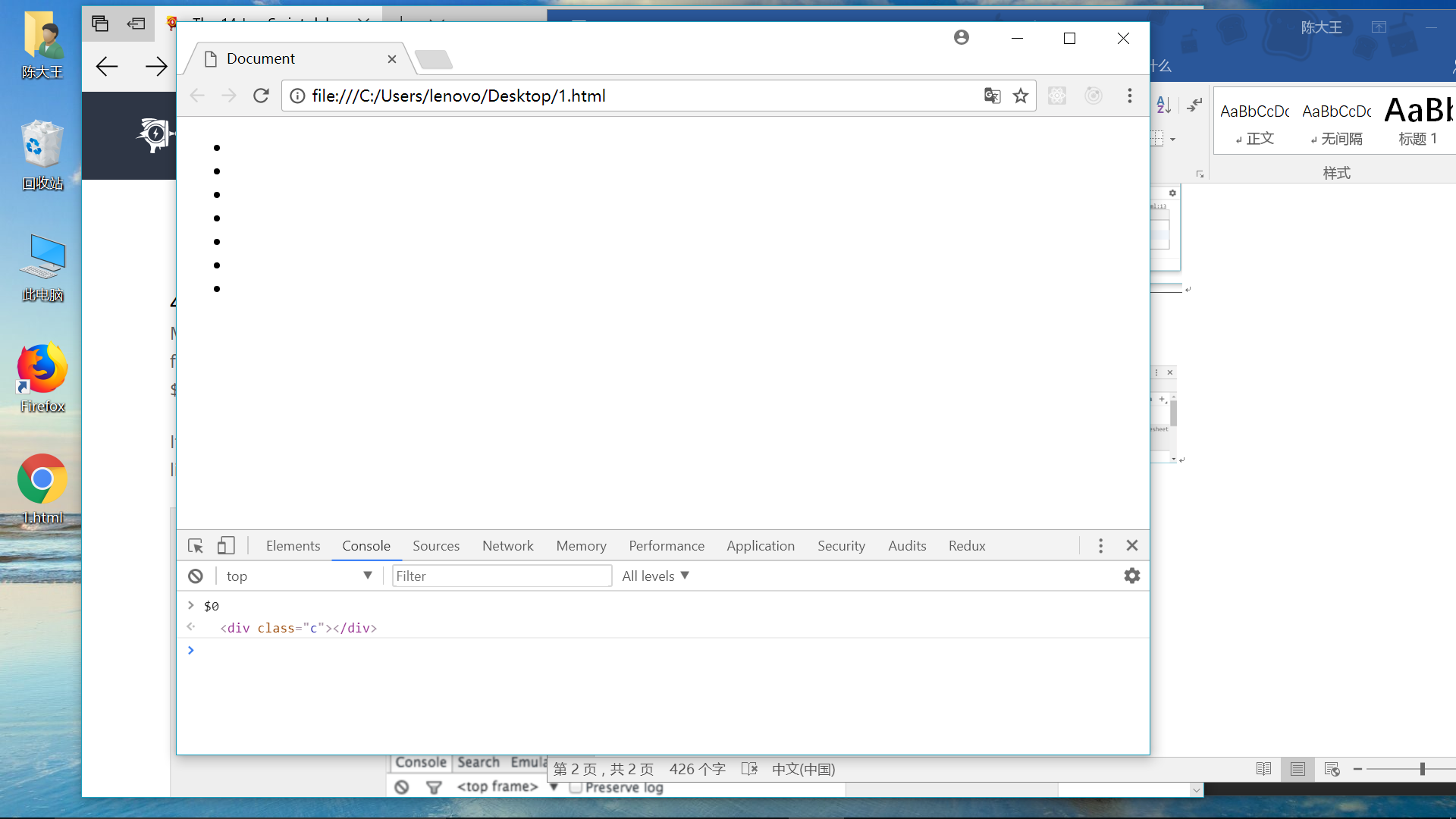
虽然桌上的每个移动设备都很棒，但在现实世界中却是不可行的。 如何调整您的视口，而不是？ Chrome为您提供所需的一切。 跳入您的检查员，然后点击“切换设备模式”按钮。 观看您的媒体查询来生活！



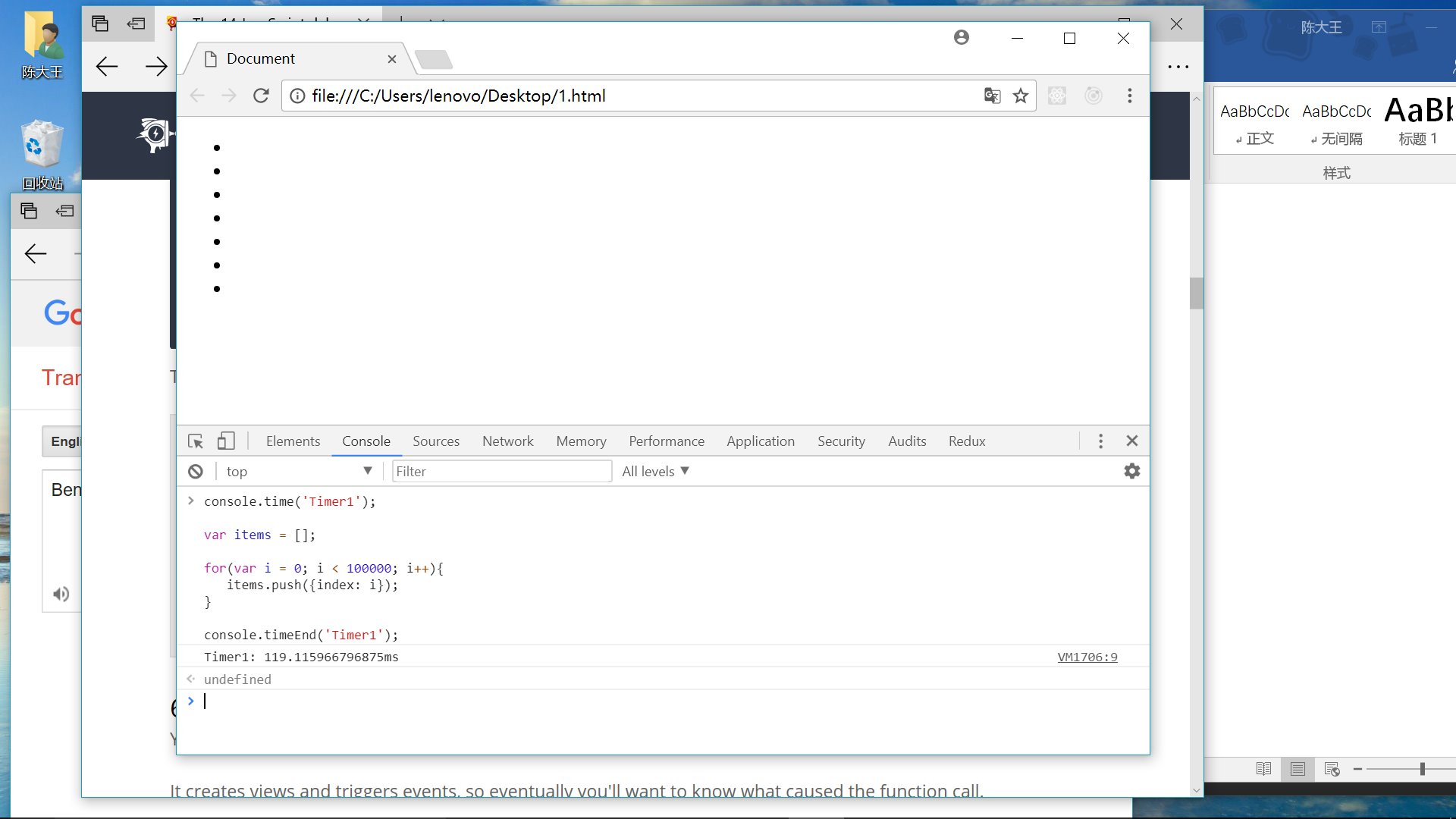
1. How to find your DOM elements quickly

$0 代表当前选中的元素





1. Benchmark loops using console.time() and console.timeEnd()



1. Get the stack trace for a function 获取函数的堆栈跟踪

var car;

var func1 = function() {

func2();

}

var func2 = function() {

func4();

}

var func3 = function() {

}

var func4 = function() {

car = new Car();

car.funcX();

}

var Car = function() {

this.brand = "volvo";

this.color = "red";

this.funcX = function() {

this.funcY();

}

this.funcY = function() {

this.funcZ();

}

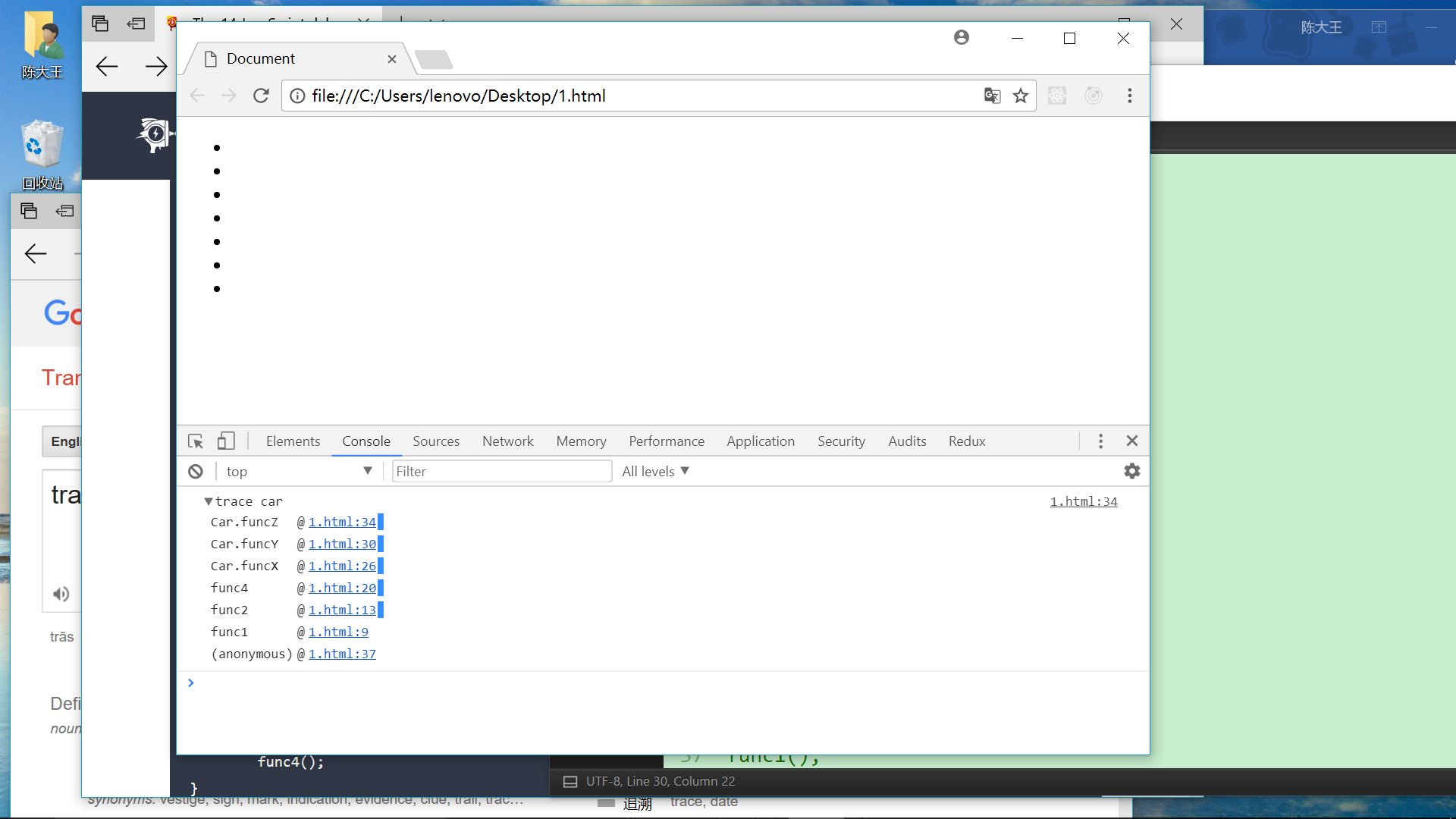
this.funcZ = function() {

console.trace("trace car")

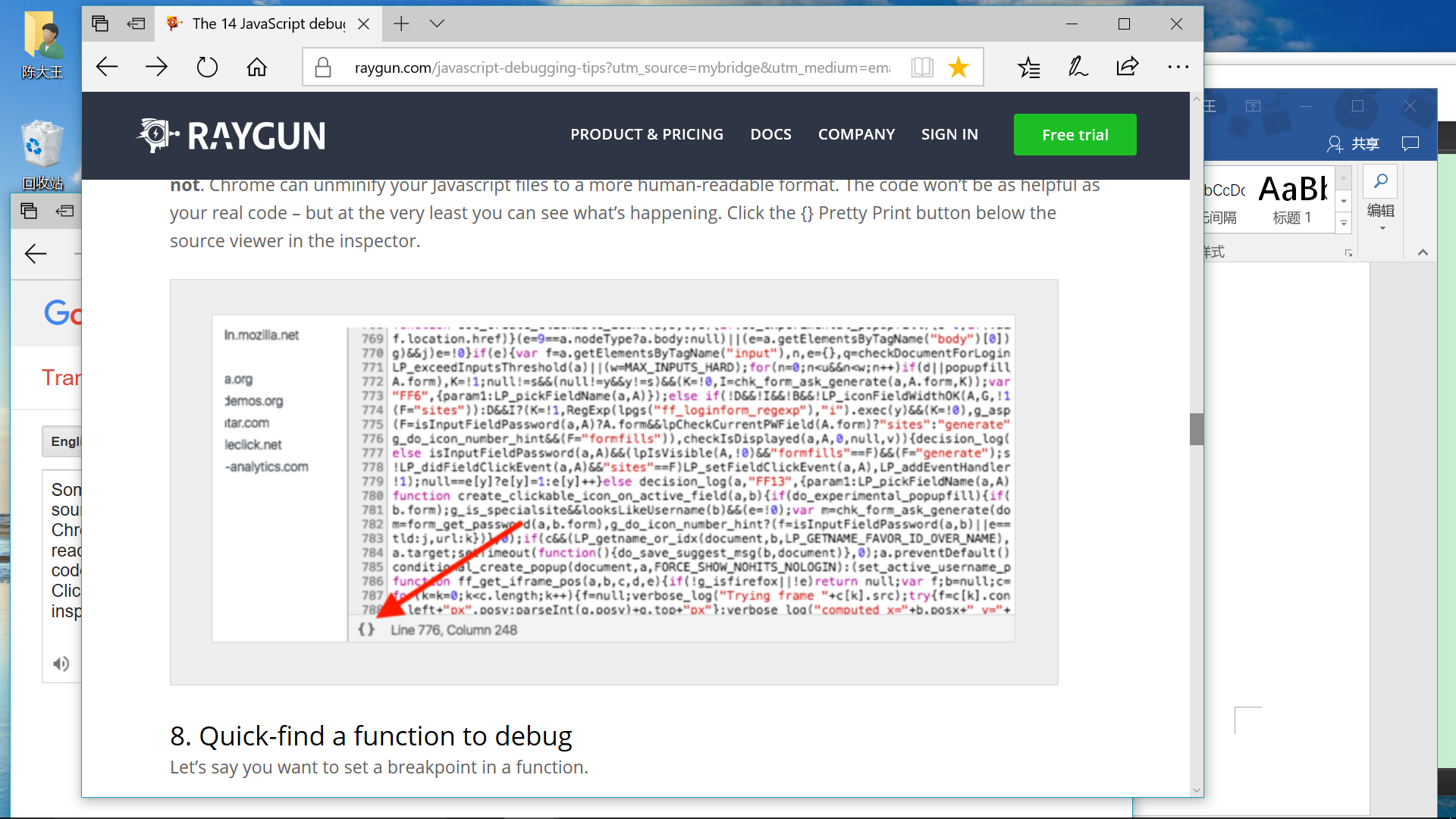
}

}

func1();



1. Unminify code as an easy way to debug JavaScript



1. Quick-find a function to debug

var func1 = function() {

func2();

};

var Car = function() {

this.funcX = function() {

this.funcY();

}

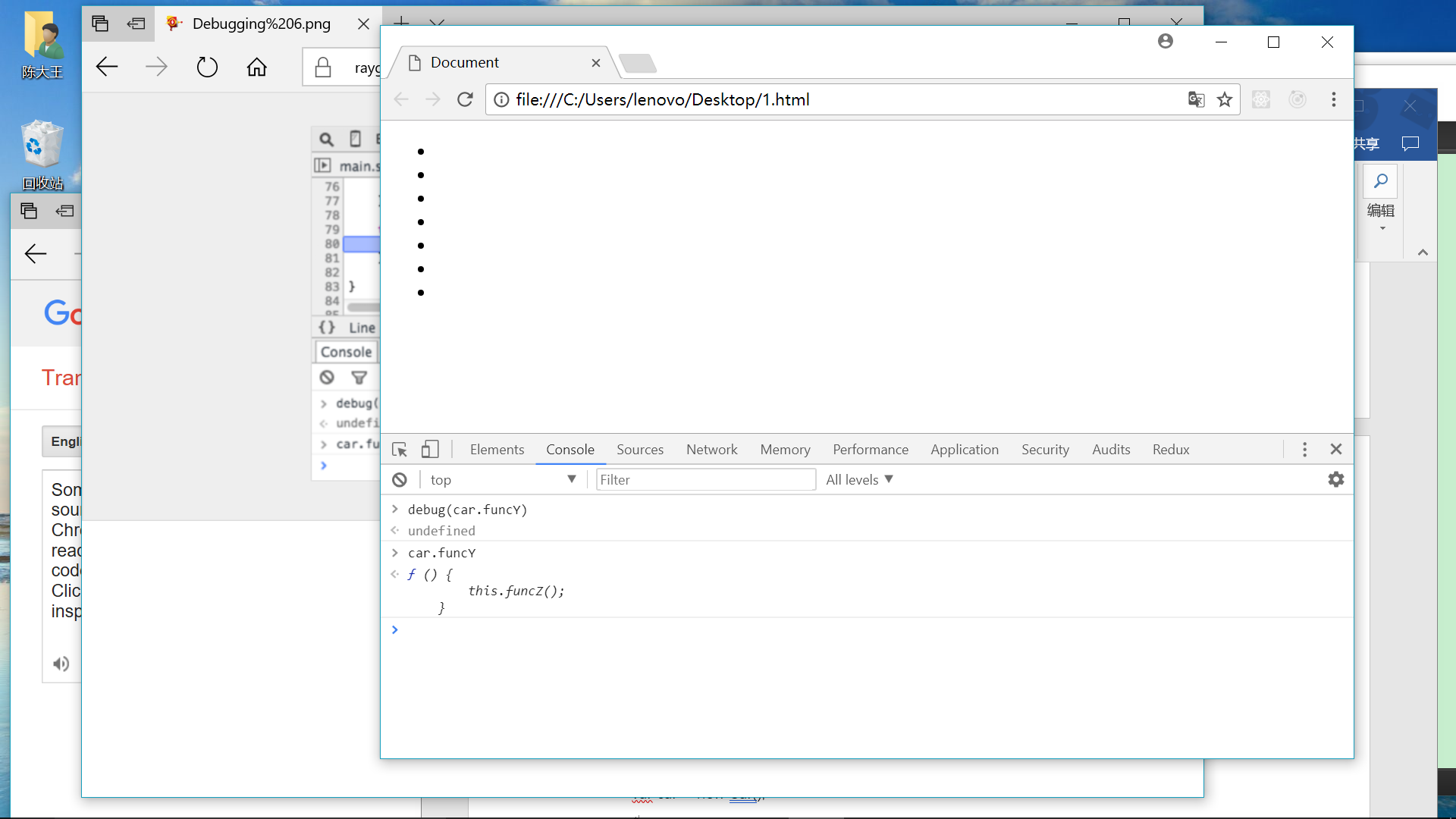
this.funcY = function() {

this.funcZ();

}

}

var car = new Car();



1. Black box scripts that are NOT relevant

Today we often have a few libraries and frameworks on our web apps. Most of them are well tested and relatively bug-free. But, the debugger still steps into all the files that have no relevance for this debugging task. The solution is to black box the script you don’t need to debug. This could also include your own scripts. Read more about debugging black box in this article.

1. Find the important things in complex debugging

console.todo = function(msg) {

console.log(' %c %s %s %s', 'color: yellow; background-color: black;', '–', msg, '–');

}

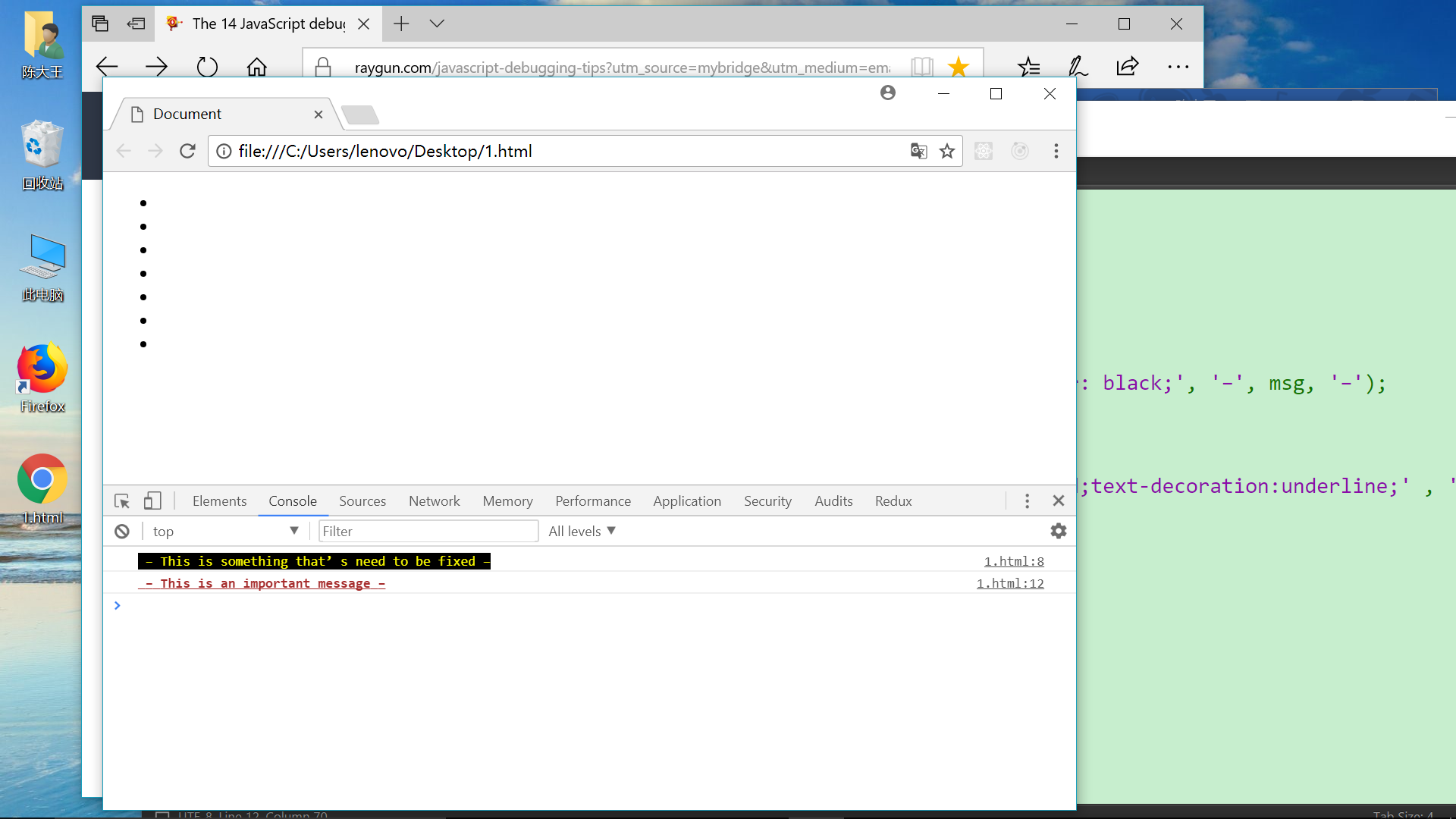
console.important = function(msg) {

console.log(' %c %s %s %s', 'color: brown; font-weight: bold;text- decoration:underline;' , '–', msg, '–');

}

console.todo("This is something that’ s need to be fixed");

console.important("This is an important message");

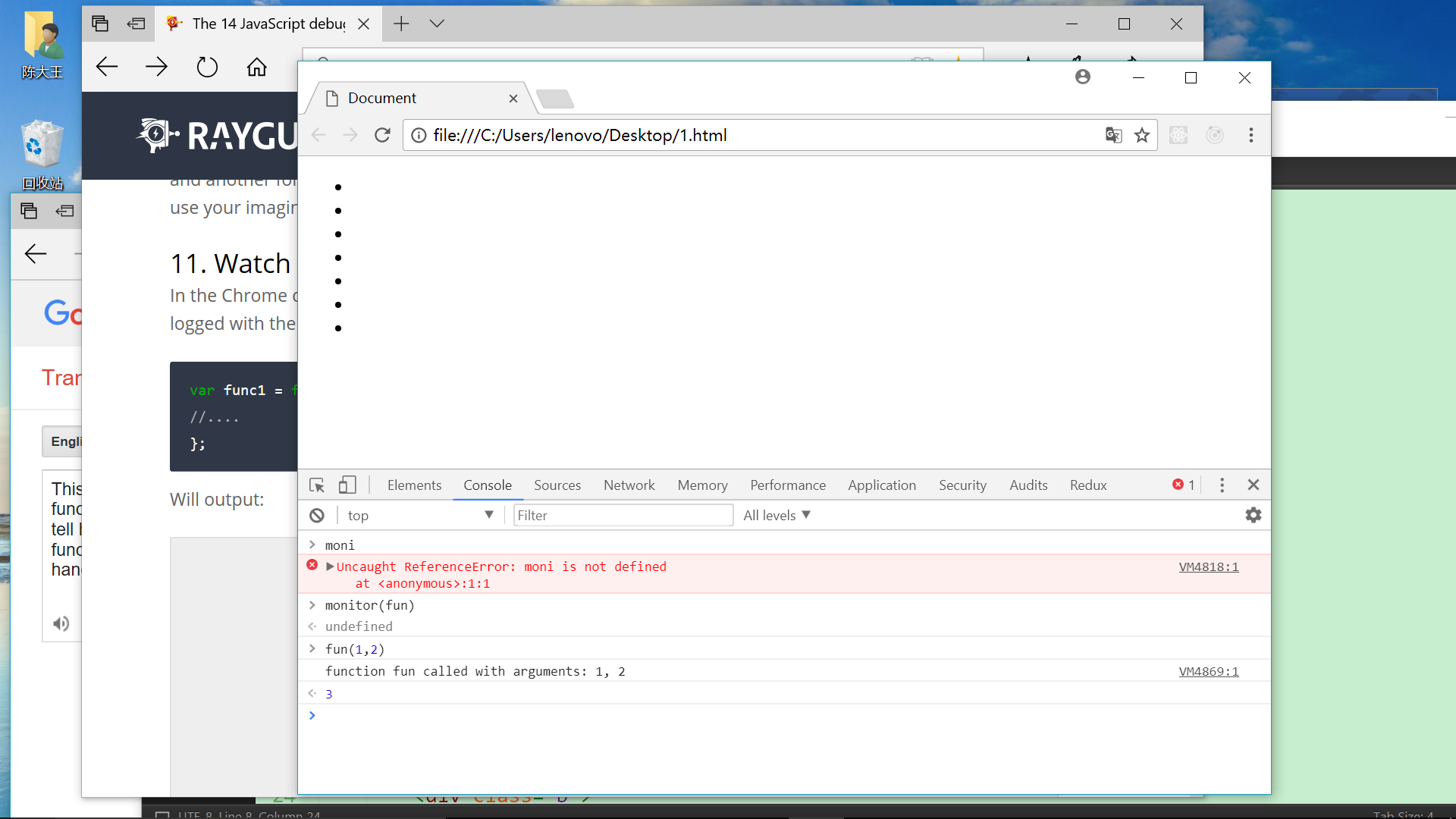


1. Watch specific function calls and its arguments

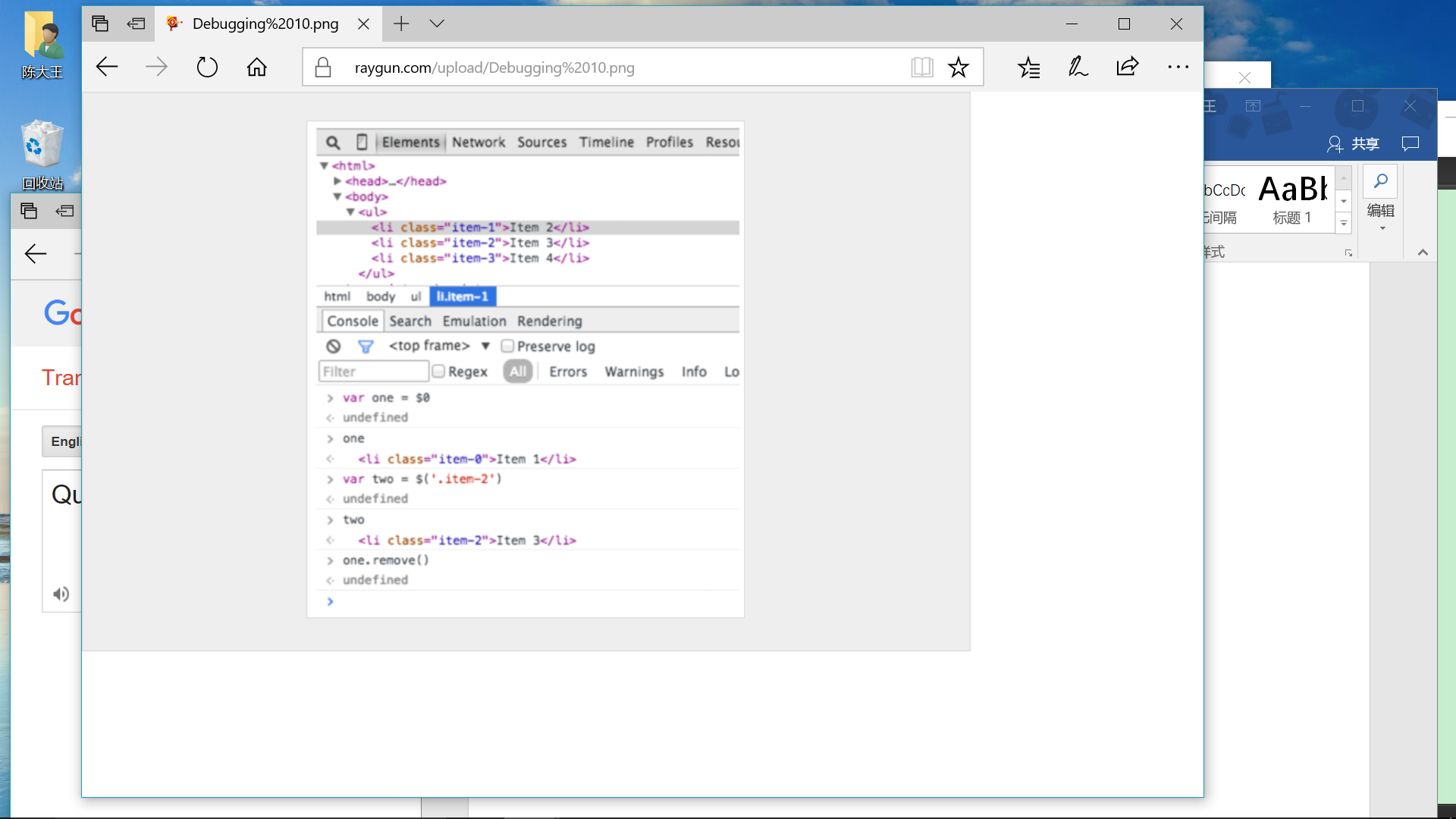
function fun(x,y){

return x+y;

}



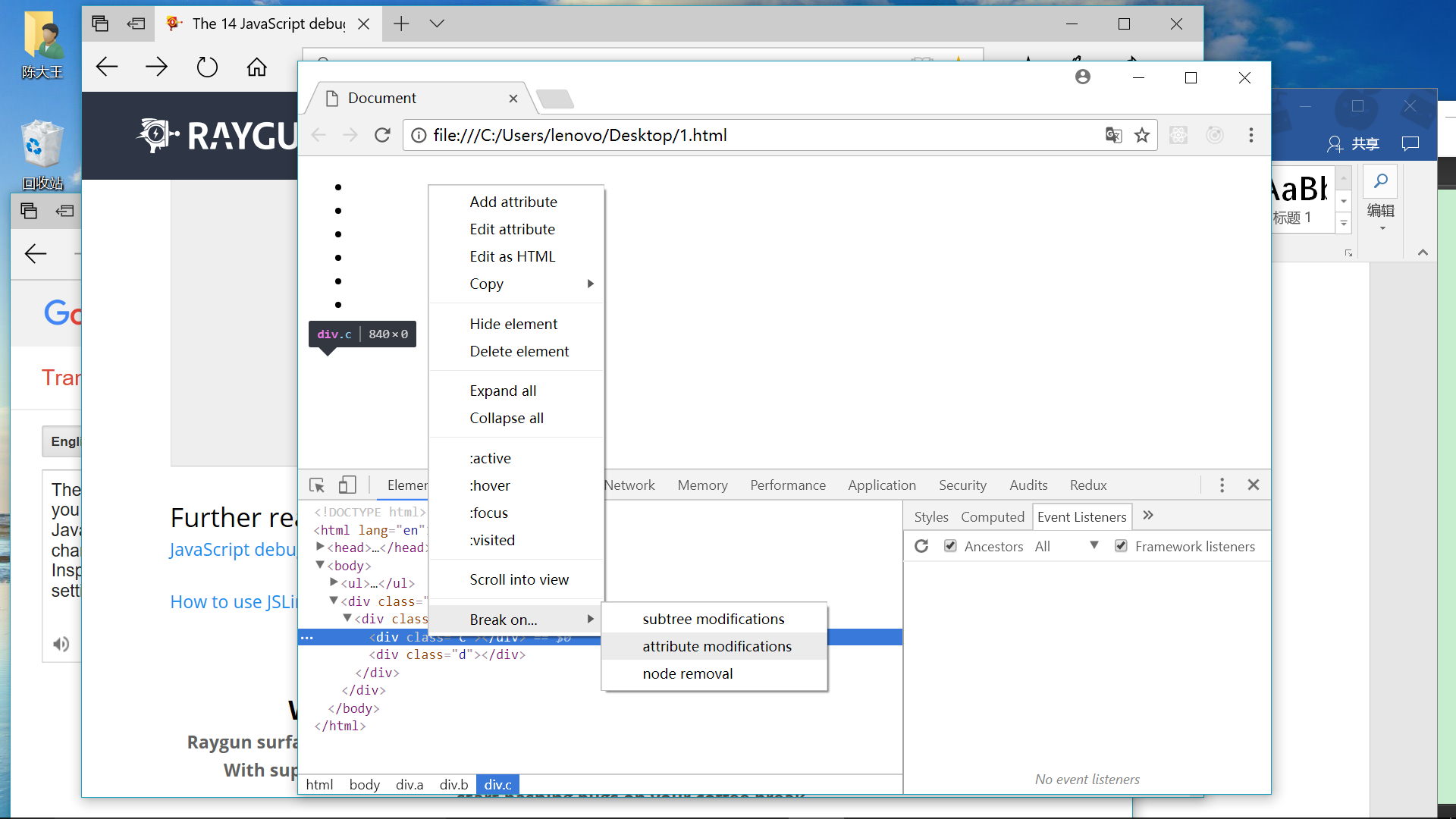
1. Quickly access elements in the console



1. Postman is great (but Firefox is faster)

Open up the inspector and go to the network tab. Right-click on the desired request and choose Edit and Resend. Now you can change anything you want. Change the header and edit your parameters and hit resend

1. Break on node change



参考网址： https://raygun.com/javascript-debugging-tips?utm\_source=mybridge&utm\_medium=email&utm\_campaign=read\_more