\*js에서 this는 선언되었을 때가 아닌, 호출되었을 때 결정된다.(동적할당)

즉 자바스크립트에서 this는 ‘누가 나를 불렀느냐’를 뜻한다.

const person = function (name) {

    this.name = name;  // 이때 this는 결정되지 않는다.

}

const person1 = new person('park');  // 이 때 결정 된다.

console.log(person1.name);  // park

1. Function(함수 선언문)에서 this

- 일반적으로 일반 함수에서 this는 전역 객체(window)를 가르킨다.

- 모든 내부함수(함수 속에 함수, 메소드 속 함수)의 this는 전역 객체(window)를 가르킨다.

- 콜백함수로 일반함수가 올 경우 전역객체(window)를 가르킨다.

function foo() {

    console.log("foo's this: ",  this);  // window

    function bar() {

      console.log("bar's this: ", this); // window

    }

    bar();

  }

  foo();

1. 메서드에서의 this 메서드 호출 시

메서드 내부 코드에서 사용된 this는 해당 메서드를 호출한 객체(person)로 바인딩 된다..

var person = {

    firstName: 'John',

    lastName: 'Doe',

    fullName: function () {

      return this.firstName + ' ' + this.lastName;

    },

  };

  console.log(person.fullName()); //"John Doe"

1. New 키워드를 사용한 인스턴스 객체에서의 this

생성자 함수가 생성하는 객체로 this가 바인딩 된다

function Person(name) {

    this.name = name;

  }

  var kim = new Person('kim');

  var lee = new Person('lee');

  console.log(kim.name); //kim

  console.log(lee.name); //lee

\*그러나 new 키워드를 빼먹는 순간 일반 함수 호출과 같아지기 때문에, 이 경우는 this가 window에 바인딩된다.

var name = 'window';

function Person(name) {

  this.name = name;

}

var kim = Person('kim');

console.log(window.name); //kim

1. 화살표 함수에서의 this(Arrow Function)

화살표 함수에서 this는 자신을 감싼 정적 범위.

화살표 함수의 경우, 자체적인 this를 가질 수 없다.

* 화살표 함수를 call(), bind(), apply()를 사용해 호출할 때 this의 값을 정해주더라도 무시한다.
* 어떤 방법을 사용하든 화살표 함수에 this는 생성 시점의 것으로 설정된다.
* 동적으로 바인딩 되는 this를 정적으로 바인딩 되게 하기 위해 등장하였다. 동적으로 결정 되는 일반 함수와는 달리 화살표 함수의 this는 언제나 상위 스코프의 this를 가리킨다.(Lexical this)

function func() {

    // 화살표 함수를 return

    return a => {

        console.log(this.a);

    };

}

var obj = {

    a: 2

};

var obj2 = {

    a: 3

}

var arrowFunc = func.call(obj);

arrowFunc.call(obj2); // 출력: 2

\*bind : bind 함수를 사용하면 this는 내가 정한 object로 고정

function f() {

    return this.a;

  }

  var g = f.bind({a: 'azerty'});

  console.log(g()); // azerty

  var h = g.bind({a: 'yoo'}); // bind는 한 번만 동작함!

  console.log(h()); // azerty

  var o = {a: 37, f: f, g: g, h: h};

  console.log(o.a, o.f(), o.g(), o.h()); // 37, 37, azerty, azerty