Javascript Functional programming

해당 글은 함수형 프로그래밍에 대한 모든 내용이 아닌 자바스크립트 언어에 대한 한정적인 내용만 담고 있습니다.

함수형 프로그래밍 개념

함수형 프로그래밍은 자료 처리를 수학적 함수의 계산으로 취급하고 상태와 가변 데이터를 멀리하는 프로그래밍 패러다임의 하나이다.

순수 함수

순수 함수 개념

순수 함수는 다음과 같은 두가지를 만족해야 한다.

- 순수 함수는 같은 인자를 받았을때, 같은 결과를 반환해야 한다.
- 부작용(side effect) 없는 함수, 즉 함수의 실행이 외부에 영향을 끼치지 않는다.

```
순수함수
```

```
Math.sin(Math.PI/2)
Math.cos(Math.PI/2)
```

장점

- 가독성과 유지 관리 편의성 향상.
  - 인수가 제공되는 경우 특정 태스크를 완료하도록 디자인되어 있음.
  - 함수는 외수 상태에 의존 하지 않음.
- 반복되는 개발이 쉽다.
  - 코드를 리팩토링하기 쉬워 디자인 변경 사항 구현에 용이
- 테스트와 디버깅이 더 쉽다.

순수 함수로 리팩토링

순수 함수로 리팩토링하여 의도하지 않은 불필요한 결과와 외부 종속성을 없앤다.

다음 코드는 객체의 property를 변경하기 때문에 순수 함수가 아니다.

객체 property를 변경하는 비순수 함수

```
var obj = {
 member: '',
 concat: function(str){
  this.member += str;
 }
}

obj.concat('one');
obj.concat('two');
console.log(obj.member);
```

다음 코드는 인수를 변경하기 때문에 순수 함수가 아니다.

인수 변경 비순수 함수

```
var obj = {
 member: 'one'
}
function concat (obj,str){
 obj.member += str;
}
concat(obj, 'two');
console.log(obj.member);
```

순수 함수 리팩토링

순수함수

```
function concat (str1, str2){
 return str1 + str2
}
console.log(concat ('one', 'two'));
```

1급 클래스 함수와 고차 함수

일급 함수

함수를 값으로 취급한다.

- 함수를 변수, 배열, 객체에 할당할 수 있다.
- 함수에 함수를 전달 할 수 있다.
- 함수에 함수를 반환 할 수 있다.

일급 함수

```
var f = function(){ return 'a' };
var array = [function(){return 'a'}, function(){return 'b'}];
var obj = {f: function(){return 'a'}};
function add(a,f){return a + f();};
add('a',f); // return 'aa'
function ff(){
 return function(){
  return 'a';
 }
}
ff()(); // return 'a'
```

고차 함수

고차 함수는 다음 중 하나 이상의 특징이 포함되어야 한다.

- 함수를 인자로 받는다.
- 결과로 함수를 반환한다.

고차함수

```
[1,2,3].reduce(function(a,b){
 return a+b
}, 0); // return 6
function getContext(){
 return this;
}
var f = getContext.bind(window);
f(); // return window
```

curry

curry

```
function curry(func, args) {
 var __method = func, args = [].slice.call(arguments, 1);
 return function() {
  return __method.apply(this, args.concat([].slice.call(arguments)));
 };
}
```

partial

partial

```
function partial(fn) {
 var args = [].slice.call(arguments, 1);
 return function () {
  var arg = 0;
  var a = args.slice();
  for ( var i = 0; arg < arguments.length; i++ )
   if(args[i] === undefined)
    a[i] = arguments[arg++];
  return fn.apply(this, a);
 }
}
```

함수 합성

두 개 이상의 함수를 합성한 합수이다. 즉 하나의 함수의 치역이 다른 함수의 정의역이 된다.

compose

compose

```
function compose(f1, f2){
 return function () {
  return f1.call(this, (f2.apply(this, arguments)));
 }
}
```

다음 코드는 함수 합성에 대한 내용이다.

함수 합성

```
function not(b){
 return !b;
}
/**
* isNumber
* @desc 인자가 숫자인지 여부를 판단한다.
* @param {*} num
* @return  {Boolean}
* @function
**/
var isNumber = compose(not, isNaN);
isNumber(1); //true
isNumber('a'): //false
```

재귀

자기 자신을 호출하는 함수

flat

```
function flat(array){
 if(Array.isArray(array))
  return array.reduce(function (arr, item){
   [].push.apply(arr, flat(item));
   return arr;
  }, []);
 else
  return [array]
}
```

함수형 프로그래밍의 문제점

너무 깊은 재귀

자바스크립트는 재귀 호출에 대해서 최적화가 되어 있지 않기 때문에 재귀 호출 에러가 발생할 수 있다. (스택 깨짐)

너무 깊은 재귀 호출

```
function even(n){
 if ( n == 0 )
  return true;
 return odd(Math.abs(n) -1);
}
function odd(n){
 if( n == 0 )
  return false;
 return even(Math.abs(n) -1);
}

even(1000000);
//RangeError: Maximum call stack size exceeded
```

트램펄린 구조를 이용해서 위 현상을 피하는 방법

트램펄린 구조는 중첩 호출을 평탄화 시킨 호출로 바꾸는 것이다.

트램펄린

```
function even(n){
 if(n == 0)
  return true;
 return odd.bind(null, Math.abs(n) - 1 );
}
function odd (n) {
 if(n == 0)
  return false;
 return even.bind(null, Math.abs(n) - 1 );
}
function trampoline(fn){
 var res = fn.call.apply(fn, arguments);
 while(typeof res === 'function'){
  res = res();
 }
 return res;
}
even(1000000);
```

명령형 프로그래밍과 함수형 프로그래밍 비교

특징	명령형 방법	함수형 방법
프로그래머가 중점을 두는 부분	작업을 수행하는 방법과 상태의 변경을 추적하는 방법	원하는 정보와 필요한 변환
상태 변경	중요	존재하지 않음
실행순서	중요	중요도가 낮음
주요 흐름 제어	루프, 조건 및 함수 호출	재귀를 비롯한 함수 호출
주요 조작 단위	클래스나 구조체의 인스턴스	1급 개체와 데이터 컬렉션인 함수

참고

http://ko.wikipedia.org/wiki/%ED%95%A8%EC%88%98%ED%98%95_%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D

http://msdn.microsoft.com/ko-kr/library/bb669144.aspx

http://msdn.microsoft.com/ko-kr/library/bb669139.aspx

http://ko.wikipedia.org/wiki/%ED%95%A9%EC%84%B1%ED%95%A8%EC%88%98