Problema I

A continuación se muestra una manera desordenada y poco eficiente de crear usuarios. Además de estos usuarios, crea al usuario2, al usuario3, al usuario4, al usuario5, con las mismas características que se muestran en la figura inmediatamente anterior. ¿Puedes mostrar estas características de los usuarios de manera ordenada y coherente en consola? ¿Tienen los usuarios alguna funcionalidad, pueden hacer algo?

Solucion:

se usa la solucion creando el siguiente codigo:

```
let user = {
    nombre: 'Paola',
    apellido: 'Ortiz',
    email: 'paola@company.ru',
    direccion: {
        municipio: 'Jocotenango',
        calle: 'Calle ancha',
        numero: 25,
    }
};
```

y se recrea el usuario hasta el número 5:

```
let user = {
    nombre: 'Paola',
    apellido: 'Ortiz',
    email: 'paola@company.ru',
    direccion: {
        municipio: 'Jocotenango',
        calle: 'Calle ancha',
        numero: 25,
    }
};
let user2 = {
    nombre: 'Carmen',
    apellido: 'Santiago',
    email: 'carmen@company.ru',
    direccion: {
        municipio: 'Guatemala',
        calle: '2 calle',
        numero: 3-21,
    }
```

```
};
let user3 = {
    nombre: 'Henry',
    apellido: 'Martinez',
    email: 'henry@company.ru',
    direccion: {
        municipio: 'Santa Catarina Pinula',
        calle: 'Esperanza',
        numero: 18,
    }
};
let user4 = {
    nombre: 'Carlos',
    apellido: 'Pineda',
    email: 'carlos@company.ru',
    direccion: {
        municipio: 'Jutiapa',
        calle: 'las orquideas',
        numero: 60,
    }
};
let user5 = {
    nombre: 'Victor',
    apellido: 'Palacios',
    email: 'Victor@company.ru',
    direccion: {
        municipio: 'Villanueva',
        calle: '3ra calle',
        numero: 90,
    }
};
```

¿Puedes mostrar estas características de los usuarios de manera ordenada y coherente en consola?
SI, de esta manera los usuarios se crean de una manera ordenada tomando en cuenta las características particulares para su utilización.

¿Tienen los usuarios alguna funcionalidad, pueden hacer algo?
En este ejercicio la utilidad de los usuarios es para contener la información.

Problema II

Crea otro código utilizando las características de los usuarios del Problema I, pero en esta ocasión agrega una funcionalidad, método recuperarClave, de la manera que se observa en la siguiente figura.

Despliega en consola a los 6 usuarios creados, observa que ahora si hay un orden y existe coherencia entre las propiedades y métodos. El resultado será la creación de 6 usuarios, pero en esta ocasión diremos que son 6 objetos. ¿Qué diferencias conceptuales observas entre el Problema I y el Problema II? ¿Para crear usuarios es más fácil y coherente la manera del Problema I o la manera en que se crean en el Problema II?

Solucion:

Se crean los usuarios utilizando las características del ejercicio I pero en este caso se define primero la función recuperar clave:

```
function recuperarClave (){
    console.log ("recuperarClave")
}
```

A cada uno de los usuarios se les agrega la función para hacer la recuperación de la clave esto le da una funcionalidad al objeto:

```
let user = {
    nombre: 'Paola',
    apellido: 'Ortiz',
    email: 'paola@company.ru',
    direccion: {
        municipio: 'Jocotenango',
        calle: 'Calle ancha',
        numero: 25,
    },
    recuperarClave:recuperarClave
};
let user2 = {
    nombre: 'Carmen',
    apellido: 'Santiago',
    email: 'carmen@company.ru',
    direccion: {
```

```
        municipio: 'Guatemala',
        calle: '2 calle',
        numero: 3-21,
    },
    recuperarClave:recuperarClave
};
let user3 = {
    nombre: 'Henry',
    apellido: 'Martinez',
    email: 'henry@company.ru',
    direccion: {
        municipio: 'Santa Catarina Pinula',
        calle: 'Esperanza',
        numero: 18,
    },
    recuperarClave:recuperarClave
};
let user4 = {
    nombre: 'Carlos',
    apellido: 'Pineda',
    email: 'carlos@company.ru',
    direccion: {
        municipio: 'Jutiapa',
        calle: 'las orquideas',
        numero: 60,
    },
    recuperarClave:recuperarClave
};
let user5 = {
    nombre: 'Victor',
    apellido: 'Palacios',
    email: 'Victor@company.ru',
    direccion: {
        municipio: 'Villanueva',
        calle: '3ra calle',
        numero: 90,
    },
    recuperarClave:recuperarClave
};
```

¿Qué diferencias conceptuales observas entre el Problema I y el Problema II?
En el problema II se asigna una función para dar una utilidad al objeto

¿Para crear usuarios es más fácil y coherente la manera del Problema I o la manera en que se crean en el Problema II?
es más fácil y coherente la creación de los usuarios de el problema II

Problema III

Crea otro código utilizando el Problema II, pero ahora agrega a cada usuario (objeto) la propiedad dpi y el método cambiar Dirección. Utiliza el ejemplo 05-Objetos/02-dinamico.js como guía para solucionar este problema.

Solución:

Se crea la funcion para cambiar la dirección utilizando los mismos usuarios uso en la consola será visto en video:

```
function recuperarClave() {
  console.log("recuperarClave");
}
function cambiarDireccion(Municipio, Calle, Numero) {
  this.direccion.municipio = Municipio;
  this.direccion.calle = Calle;
  this.direccion.numero = Numero;
}

let user = {
  nombre: "Paola",
  apellido: "Ortiz",
  dpi: "2938495869384",
  email: "paola@company.ru",
  direccion: {
    municipio: "Jocotenango",
    calle: "Calle ancha",
    numero: 25,
  },
  recuperarClave: recuperarClave,
  cambiarDireccion: cambiarDireccion,
};
let user2 = {
  nombre: "Carmen",
  apellido: "Santiago",
  dpi: "299848573748128",
  email: "carmen@company.ru",
  direccion: {
```

```
    municipio: "Guatemala",
    calle: "2 calle",
    numero: 3 - 21,
  },
  recuperarClave: recuperarClave,
  cambiarDireccion: cambiarDireccion,
};
let user3 = {
  nombre: "Henry",
  apellido: "Martinez",
  dpi: "299848573756789",
  email: "henry@company.ru",
  direccion: {
    municipio: "Santa Catarina Pinula",
    calle: "Esperanza",
    numero: 18,
  },
  recuperarClave: recuperarClave,
  cambiarDireccion: cambiarDireccion,
};
let user4 = {
  nombre: "Carlos",
  apellido: "Pineda",
  dpi: "299845721356789",
  email: "carlos@company.ru",
  direccion: {
    municipio: "Jutiapa",
    calle: "las orquideas",
    numero: 60,
  },
  recuperarClave: recuperarClave,
  cambiarDireccion: cambiarDireccion,
};
let user5 = {
  nombre: "Victor",
  apellido: "Palacios",
  dpi: "2876548573756789",
  email: "Victor@company.ru",
  direccion: {
    municipio: "Villanueva",
    calle: "3ra calle",
    numero: 90,
  },
```

```
  recuperarClave: recuperarClave,
  cambiarDireccion: cambiarDireccion,
};
```

Problema IV

Utilizando las propiedades y métodos agrupados en cada usuario creado en el Problema III, vuelve a crear esos usuarios (objetos) pero en esta ocasión crea los usuarios (objetos) utilizando una factory function.

Solución:

Se integra la funcionalidad de creación de usuario por medio de la función create user de una manera que permita la creación de usuario de una manera más resumida y amigable:

```
function createUser(
  Nombre,
  Apellido,
  Dpi,
  Email,
  DireccionMunicipio,
  DireccionCalle,
  DireccionNumero
) {
  return {
    nombre: Nombre,
    apellido: Apellido,
    dpi: Dpi,
    email: Email,
    direccion: {
      municipio: DireccionMunicipio,
      calle: DireccionCalle,
      numero: DireccionNumero,
    },
    recuperarClave: recuperarClave,
    cambiarDireccion: cambiarDireccion,
  };
}
let user6 =
createUser("Juan","Castro","2309458976456","juan@gmail.com","Zacapa","tamarindo","65")
```