

Singleton Factory

Clase que implementa el patrón Factory con un diccionario genérico

Conserva un diccionario estático de elementos T y los instancia una sola vez.

Declaration:

```
SingletonFactoryArray<Person> personList = new SingletonFactoryArray<Person>();
```

Agregado de clases ya creadas:

```
Person p1 = new Person();  
p1.Id = 1000;  
p1.Nombre = "Gaus";  
personList.Add(p1.Nombre, p1);  
  
Person p2 = new Person();  
p2.Id = 2000;  
p2.Nombre = "Hendryx";  
personList.Add(p2.Nombre, p2);
```

Por lo tanto personList contendrá en su diccionario interno clases únicas de tipo Person

Identificador	Objet
Gaus	p1.Id = 1000; p1.Nombre = "Gaus";
Hendryx	p2.Id = 2000; p2.Nombre = "Hendryx";

Creación automática de instancia

Para instanciar automáticamente una clase con el patrón Factory simplemente hay que llamar al método Create con el parámetro key

Esta sobrecarga utiliza el constructor de Person() . por lo tanto no inicializa ninguna propiedad de la clase

```
Person p3 = personList.Create("Maria");
```

Esta sobrecarga utiliza el constructor de Person(int id,string nombre) y permite inicializar las propiedades de la clase.

```
public Person(int id,string nombre) {  
    this.Id = id;  
    this.Nombre = nombre;  
}
```

```
Person p4 = personList.Create("Robert", new Object[] { 3000, "Robert" });
```

Por lo tanto personList contendrá en su diccionario interno clases únicas de tipo Person

Identificador	Objet
Gaus	p1.Id = 1000; p1.Nombre = "Gaus";
Hendryx	p2.Id = 2000; p2.Nombre = "Hendryx";
"Maria"	P3.Id = 0 P3.Nombre = ""
Robert	p1.Id = 3000; p1.Nombre = " Robert ";

Eliminación de objetos:

```
personList.Remove("Robert");  
personList.Remove("Gaus");
```

Eliminar todos los objetos

```
personList.Clear();
```