Tweety por dentro

La clase Sistema, es un singleton, es la encargada de manejar las altas/bajas de usuarios, como así también el archivado de estos para backup de las cuentas con sus mensajes y cambios.

En bien se carga el parcel se crea un archivo 'tweetyUserList.txt' en el directorio de la imagen de VW, este archivo es manejado o manipulado por la clase ListaUsuarios, que es la encargada de guardar las cuentas, de buscar cuentas, borrarlas, y también de hacer un volcado de estas al archivo 'tweetyUserList.txt', esta clase esta en estrecha relación con la clase Sistema (la idea inicial que tuve con esta clase es poder usarla de interfaz entre el sistema y el medio que sea de archivado de las cuentas).
Sistema es el que se encarga de poner las políticas, como por ejemplo no permitir dos cuentas con mismo nombre, validación de una cuenta, logeo de la misma, o borrado de esta.
Para garantizar que el que use la clase Sistema es alguien de confianza esta la clase Administrador, que es simplemente una "validadora" del usuario de Sistema(Administrador también es Singleton).

La clase Usuario es la que instanciada contiene todo lo que se necesita para poder enviar/recibir mensajes como también agregar/eliminar contactos y agruparlos en categorías a afines, se puede decir que es la que representa al usuario en el sistema este.

La clase grupo es una clase auxiliar de Usuario, esta está para poner reglas y orden en el manejo del grupo de contactos, además de un nombre para distinguirse de otros grupos

La clase mensaje es la que le da el formato al texto que un usuario esta queriendo enviar a sus contactos, este formato le pone una fecha que no puede ser cambiado una vez creado y le pone el nick de quien creo ese mensaje, que tampoco puede ser cambiado una vez creada la instancia. Este mensaje es el que se le envía a los contactos del usuario que lo creo.

Sistema>
```
Smalltalk.Core defineClass: #Sistema
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: 'myUsers '
	classInstanceVariableNames: 'sistema '
	imports: ''
	category: ''
```

Metodos de clase
>getInstance

```
	sistema isNil ifTrue: [sistema := super new initialize].
	^sistema
```

>postLoadActionFor: unParcel

```
	| aux |
	aux := BinaryObjectStorage onNew: 'TweetyUsersList.txt' asFilename writeStream.
	[aux nextPut: OrderedCollection new dcopy] ensure: [aux close].
	^self
```

Metodos de instancias
>initialize

```
	myUsers := ListaUsuarios new.
	^self
```

>agregar: unUser

```
	(myUsers existeNick: unUser nick)
		ifFalse:
			[myUsers agregar: unUser.
			self changed: #agregarUser].
	^self
```

>buscar: unNick

```
	| u |
	(u := myUsers seleccionar: [:usuario | usuario nick = unNick]) isEmpty ifTrue: [^nil].
	^u first
```

>buscarPorDato: unDato

```
	^myUsers seleccionar: [:usuario | (usuario datos findString: unDato startingAt: 1) > 0]
```

>buscarPorNombre: unNombre

```
	^myUsers seleccionar: [:usuario | (usuario nombre findString: unNombre startingAt: 1) > 0]
```

>eliminar: unUser validadoPor: unaClave

```
	((Administrador getInstance: unaClave) notNil and: [myUsers existeNick: unUser nick])
		ifTrue:
			[unUser listaContactos do: [:cont | cont listaSiguiendo remove: unUser
ifAbsent: [nil]].
			unUser listaSiguiendo do: [:cont | cont listaContactos remove: unUser
ifAbsent: [nil]].
			myUsers borrar: unUser.
			self changed: #eliminarUser].
	^self
```

```
>existe: unNick

        ^myUsers existeNick: unNick

>login: unNick pass: unaPass

      |u|
          u:= (myUsers seleccionar:[:user | user nick = unNick]) first .
       (u esClave: unaPass) ifTrue:[^u]
                                    ifFalse:[^nil].

>users

        ^myUsers
```

ListaUsuarios>

```
Smalltalk.Core defineClass: #ListaUsuarios
        superclass: #{Core.Object}
        indexedType: #none
        private: false
        instanceVariableNames: 'misUsuarios '
        classInstanceVariableNames: ''
        imports: ''
        category: ''
```

Metodos de clase

```
>new

        ^super new initialize
```

Metodos de instancias

```
>initialize

        | boss |
        boss := BinaryObjectStorage onOld: 'TweetyUsersList.txt' asFilename readStream.
        [misUsuarios := boss next] ensure: [boss close].
        ^self


>agregar: unUsuario

        misUsuarios add: unUsuario.
        self guardarLista.
        ^self


>borrar: unUsuario

        misUsuarios remove: unUsuario ifAbsent: [^self].
        self guardarLista.
        ^self
```

```
>existe: unUsuario

	^misUsuarios includes: unUsuario


>existeNick: unNick

	^(misUsuarios select: [:unUsuario | unUsuario nick = unNick]) notEmpty


>existeNombre: unNombre

	^misUsuarios select: [:unUsuario | unUsuario nombre = unNombre]


>guardarLista

	| boss |
	boss := BinaryObjectStorage onNew: 'TweetyUsersList.txt' asFilename writeStream.
	[boss nextPut: misUsuarios dcopy] ensure: [boss close].
	^self


>misUsuarios

	^misUsuarios


>misUsuarios: anUsers

	misUsuarios := anUsers.
	^self


>seleccionar: aBlock

	^misUsuarios select: aBlock
```

Administrador>

```
Smalltalk.Core defineClass: #Administrador
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: 'clave '
	classInstanceVariableNames: 'admin '
	imports: ''
	category: ''
```

Metodos de clase

>getInstance: unaClave

```
admin isNil ifTrue: [admin := super new initialize].
admin clave = unaClave ifTrue: [^admin].
^nil
```

Metodos de instancias

>initialize

```
clave := 'tweety'.
^self
```

>clave

```
^clave
```

>clave: unaClave

```
clave := unaClave.
^self
```

Usuario>

```
Smalltalk.Core defineClass: #Usuario
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: 'nick nombre datosp listaEnviados listaRecibidos listaGrupos
listaContactos listaSiguiendo clave '
	classInstanceVariableNames: ''
	imports: ''
	category: ''
```

Metodos de clase

>new: unNick nombre: unNombre datos: unD clave: unP

```
"	***This is decompiled code.***
The source was unavailable because the source pointer appears to point to
an incorrect position in the file. The file may have been modified after this
method was updated."

^super new
	initialize: unNick
	nombre: unNombre
	datos: unD
	clave: unP
```

Metodos de instancias

>initialize: unNick nombre: unNombre datos: unD clave: unaP

```
	nick := unNick.
	nombre := unNombre.
	datosp := unD.
	listaEnviados := OrderedCollection new.
	listaRecibidos := OrderedCollection new.
	listaGrupos := Dictionary new.
	listaContactos := Set new.
	listaSiguiendo := Set new.
	clave := unaP.
	^self


>agregar: unContacto en: unGrupo

	unGrupo agregarContacto: unContacto.
	^self


>agregarA: unContacto

	listaSiguiendo add: unContacto.
	unContacto agregarContacto: self.
	self changed: #agregarASiguiendo.
	^self


>agregarContacto: unContacto                    (este es un metodo privado de la instancia)

	(listaContactos includes: unContacto)
		ifFalse:
			[listaContactos add: unContacto.
			self changed: #agregarContacto].
	^self


>eliminarDe: unContacto

	listaSiguiendo remove: unContacto ifAbsent: [nil].
	unContacto eliminarContacto: self.
	self changed: #eliminarDeSiguiendo.
	^self


>eliminarContacto: unContacto                    (este es un metodo privado de la instancia)

	listaContactos remove: unContacto ifAbsent: [nil].
	listaGrupos do: [:g | g removerContacto: unContacto].
	self changed: #eliminarContacto.
	^self
```

```
>clave: unaPass

	clave := unaPass.
	^self


>crearGrupo: unString

	| g |
	g := Grupo new: unString.
	listaGrupos at: unString put: g.
	self changed: #nuevoGrupo.
	^g


>eliminarGrupo: unGrupo

	listaGrupos removeKey: unGrupo nombre ifAbsent: [nil].
	self changed: #grupoBorrado.
	^self



>datos

	^datosp


>datos: unaData

	datosp := unaData.
	^self


>esClave: anObject

	^clave = anObject


>listaContactos

	^listaContactos


>listaEnviados

	^listaEnviados

>listaEnviados: unMensaje

	listaEnviados add: unMensaje.
	^self
```

```
>listaGrupos

	^listaGrupos asOrderedCollection


>listaRecibidos

	^listaRecibidos


>listaRecibidos: unMensaje

	listaRecibidos add: unMensaje.
	^self


>listaSiguiendo

	^listaSiguiendo


>nick
	^nick


>nombre
	^nombre


>nombre: unNombre

	nombre := unNombre.
	^self


>quitar: unContacto de: unGrupo

	unGrupo eliminarContacto: unContacto.
	^self


>crearMensaje: unTexto
	^Mensaje new: self con: unTexto

>enviar: unTexto

	| m |
	m := self crearMensaje: unTexto.
	listaContactos do: [:contacto | contacto recibir: m].
	listaEnviados add: m.
	self changed: #mensajesEnviados.
	^self
```

```
>enviar: unTexto a: unGrupo

	| m |
	m := self crearMensaje: unTexto.
	unGrupo listaContactos do: [:contacto | contacto recibir: m].
	listaEnviados add: m.
	self changed: #mensajesEnviados.
	^self


>recibir: unMensaje

	listaRecibidos add: unMensaje.
	self changed: #mensajesRecibidos.
	^self
```

Grupo>

```
Smalltalk.Core defineClass: #Grupo
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: 'nombre listaContactos '
	classInstanceVariableNames: ''
	imports: ''
	category: ''
```

Metodo de instancias

```
>initialize: unNombre

	listaContactos := Set new.
	nombre := unNombre.
	^self


>agregarContacto: unContacto

	listaContactos add: unContacto.
	self changed: #nuevoContacto.
	^self

>listaContactos
	^listaContactos

>listaContactos: unaLista

	listaContactos := unaLista.
	^self

>nombre
	^nombre
```

>nombre: unNombre

```
nombre := unNombre.
^self
```

>removerContacto: unContacto

```
listaContactos remove: unContacto ifAbsent: [nil].
self changed: #contactoBorrado.
^self
```

Mensaje>

```
Smalltalk.Core defineClass: #Mensaje
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: 'duenio fecha cuerpo '
	classInstanceVariableNames: ''
	imports: ''
	category: ''
```

Metodos de clase

>new: unDuenio con: unTexto

```
^super new initialize: unDuenio con: unTexto
```

Metodos de instancias

>initialize: unDuenio con: unTexto

```
duenio := unDuenio.
fecha := Date dateAndTimeNow.
cuerpo := unTexto.
^self
```

>cuerpo

```
^cuerpo
```

>cuerpo: unTexto

```
cuerpo := unTexto.
^self
```

>duenio

```
^duenio
```

```
>fecha

	^fecha

>printOn: t1

	t1
		nextPutAll: '@';
		nextPutAll: self duenio nick;
		nextPutAll: ' - ';
		nextPutAll: self fecha printString;
		nextPutAll: ' > ';
		nextPutAll: self cuerpo.
	^self
```
